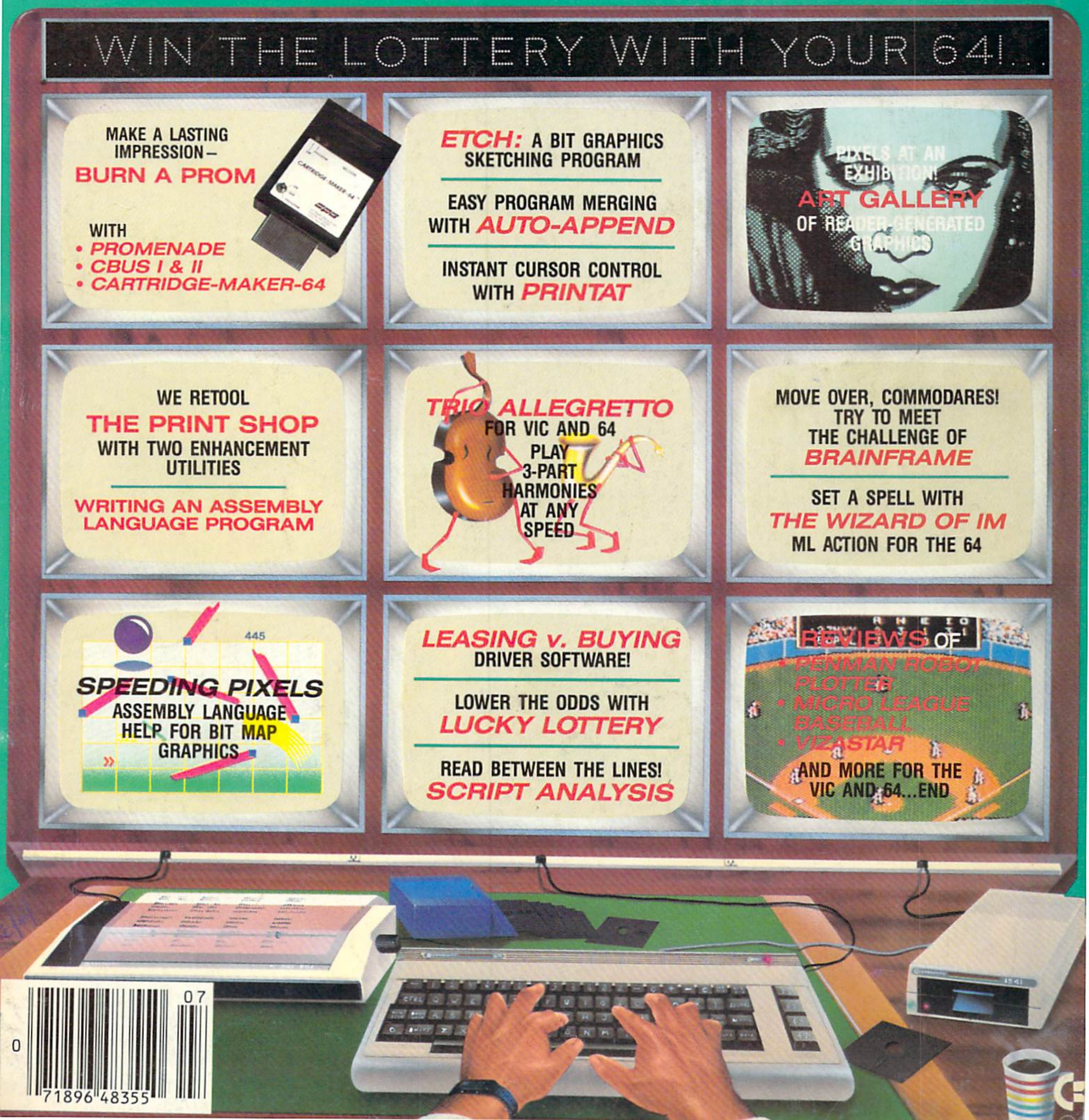
COMMODORE USERS

48355
K

Ahoy!

$2.50/ CAN. $3.00 JULY 1985



Ahoy!

CONTENTS

DEPARTMENTS

***A View from the Bridge...**of the July issue of* Ahoy!	4
***Scuttlebutt**...news from across the Commodore nation.*	7
***Errata**...if your version of* DSKDU *didn't, here's why.*	16
***Flotsam**...our readers do the writing for a change.*	46
***Art Gallery**...our readers draw on their 64's.*	54
***Reviews...**we strip-search the newest hardware and software.*	63
***S.O.S.**...a lifesaver for the puzzled programmer.*	74
***Commodares...**ranging from brain-teasers to brain-tormentors.*	81
***Program Listings...**a stellar lineup of software.*	85

FEATURES

***Trio Allegretto for VIC and 64** by Orson Scott Card**	18
***PROM Programming Made Easy** by Morton Kevelson*	27
***Rupert Report...**Dale runs down some* Speeding Pixels.**	37
The Print Shop** by Morton Kevelson*	50
Writing & Running an Assembly Program** by Mark Andrews**	77

*Includes programs: *Song Maker, Song Loader, Summertime* for the C-64; *VIC Music* for the VIC 20
**Includes programs: *CLRSCR.O* and *PLOT.$C.S* for the C-64
***Includes programs: *Print Shop to Bit Map Convertor* and *Screen Magic to DOODLE! Convertor* for the C-64
****Includes program: *Flash* for the C-64

PROGRAMS

***Script Analysis** for the C-64 by Bob Spirko*	17
***Brainframe** for the C-64 by Norm Vogel*	45
***Printat** for the C-64 by Simon Edgeworth*	45
***The Wizard of Im** for the C-64 by Bob Spirko*	47
***Lucky Lottery** for the C-64 by Bob Lloret*	53
***Auto-Append** for the C-64 and VIC 20 by William V. Braun*	62
***Etch** for the C-64 by Roger S. Macomber*	73
***Autos: Leasing v. Buying** for the C-64 by Gil Ragan*	75
***Bug Repellents** for the VIC & 64 by Kleinert and Barron*	86
***Flankspeed** for the C-64 by Gordon F. Wheat*	86

Cover art by James Regan; art inside screens by Wayne Schmidt, James Regan, Javier Romero; photography by Morton Kevelson

ISSUE NO. 19 JULY 1985

***Ahoy!* (#8750-4383) is published monthly by Ion International Inc., 45 W. 34th St., Suite 407, New York, NY, 10001. Subscription rate: 12 issues for $19.95, 24 issues for $37.95 (Canada $26.95 and $49.95 respectively). Second class postage paid at New York, NY 10001 and additional mailing offices.**

Postmaster, send address changes to *Ahoy!*, 45 W. 34th Street, Suite 407, New York, NY 10001. Direct all address changes or matters concerning your subscription to *Ahoy!*, P.O. Box #341, Mt. Morris, IL 61054. All editorial inquiries and software and hardware to be reviewed should be sent to *Ahoy!*, 45 W. 34th St., Suite 407, New York, NY 10001.

VIEW FROM THE BRIDGE

Before we talk about what's in this issue of *Ahoy!*, let's talk about what's *not* in it. Two other Commodore magazines, you may have noticed, ran reviews of the Commodore 128 Personal Computer in their June issues. Why did *Ahoy!*, which always leads the pack in announcing and reviewing new products, allow itself to be scooped in this singularly important instance?

Tina Turner summed it up best when she said, "We never ever do nothing nice and easy." We weren't about to prepare a feature article on the 128 based on an afternoon's experimentation with the machine at Commodore's headquarters, as our two competitors did. We published our surface analysis of the 128 in April; not until Commodore shipped us a model of our own did Morton Kevelson sit down for a several-week round of testing the machine inside and out. The first pieces based on that research appear in the August *Ahoy!* As anxious as you are to read the definitive review of the Commodore 128, we know a month is a long time to wait. But we've packed the July *Ahoy!* full of programs and features that should help make that month pass quickly:

• The graphics theme begun last issue continues with Part II of Roger Macomber's tutorial on *Screen Dumping on the Commodore 64*, providing an onscreen Etch-a- Sketch (turn to page 73). Then, Morton Kevelson takes an extended look at *The Print Shop*, certainly one of the most popular Commodore programs of all time. Morton and Michael Beutjer (author of last issue's *Quad-Print*) each contribute an enhancement utility that will help users get even more out of the program. (Turn to page 50.) Finally, our *Art Gallery* debuts this issue, with some of the first submissions of reader-generated graphics. (Turn to page 54.)

• Once again, Orson Scott Card's *Creating Your Own Games on the VIC and 64* transcends the genre of computer games—and published software. *Trio Allegretto* supplies machine language routines that you can imbed in your game program to play music that the program loads into BASIC strings, at any speed, on the VIC or 64. (Turn to page 18.)

• They used to call it ROM burning, back when ROMs were programmed by burning open a fusable link. The days of burning ROMs have vanished, along with the fusable links by which they were programmed; but the home user can still place his programs in Permanent Read Only Memory. In *PROM Programming Made Easy*, Morton the K shows 64 and VIC users how to do that at surprisingly low cost. (Turn to page 27.)

• Bob Lloret returns to *Ahoy!*'s program pages with *Lucky Lottery* for the 64. It's as valuable a system as you'll find anywhere for picking the winning numbers in your state lottery—which means, completely worthless. But you'll have fun predicting winners based on past weeks' results. (Turn to page 53.)

• Moving up to the pseudo-scientific, Bob Spirko's *Script Analysis* will allow 64 users to prepare a character profile based on a subject's handwriting sample. (Turn to page 17.) Bob also contributes *The Wizard of Im*, a machine language game for the 64. While the game's format is nothing new, what Bob does with a familiar theme is...magic! (Turn to page 47.)

• *Printat* by Simon Edgeworth simulates the cursor-controlling command of the same name found in some advanced versions of BASIC. (Turn to page 45.)

• William V. Braun's *Auto-Append* will allow VIC and 64 users to merge programs instantly. (Turn to page 62.)

• You've heard that it can be cheaper to lease a new car than buy one—but who can assess all the variables involved and make a determination? *You* can—with Gil Ragan's *Autos: Leasing v. Buying* for the C-64. (Turn to page 75.)

• Have Dale Rupert and Mark Andrews got big plans for you this month! In the *Rupert Report* on *Speeding Pixels*, you and Dale will develop a pair of assembly language routines that will speed up the manipulation of the 64,000 pixels in hi-res bit mapped graphics on the 64. (Turn to page 37.) In *Commodore Roots*, Mark Andrews steps you up to *Writing and Running an Assembly Language Program*. (Turn to page 77.)

At this point, we'll draw an exhausted breath and let you discover the rest of the programs and features in the July *Ahoy!* for yourself. (Didn't we *tell* you we'd make you forget all about next month's feature on the 128?) *—David Allikas*

AHOY! BULLETIN BOARD: 718-383-8909
If your computer is equipped with a modem, you can call *Ahoy!*'s BBS 24 hours a day, 7 days a week to exchange mail with other Commodore users and download information on upcoming issues of *Ahoy!*, late-breaking Commodore news, program corrections, and more.

An open letter to the readers of Ahoy Magazine
Vincent Kurek
President: The Ennon Corporation

My purpose in writing is to ask you to join me in shaping the future of the new and most unusual field in computer technology today: Artificial Intelligence.

This incredible power and spectacular creative potential are available to you, for your computer right now. However, there is an alarming possibility that such amazing technology which you have every right to, may not be available to you other that through this offer.

This is unfortunate but somewhat understandable due to the way technology is created. You see, only the business oriented corporation can finance research. It therefore is in a position to dictate immediate research goals. These goals are increasing profits through more efficient production. While valid, they are merely creative and do absolutely nothing to foster exploration in new applications. The result: technology is never used to its fullest potential. But what's worst of all is that these competitve corporations have absolutely no desire to share technology with each other, let alone with you. So, they don't. As a result, the infinitesimal amount of technology that finally trickles down to you is:

A. So expensive you are prohibited from procurring it
B. Shamefully inferior to the real thing

remember...you can buy high-tech consumer goods, but never the technology that creates it.

This same situation confronts you in the new Artificial Intelligence field, but with a difference: There is no true Artificial Intelligence for the home computer user! The few programs claiming to be Artificial Intelligence are really simulators. They are not the real thing. Possessing a mere token of the power and versatility, simulators are clearly not worth their expensive price.

I have tried repeatedly to convince my colleagues that it is in their best interest to release genuine Artificial Intelligence to the general public. The refinement, modification and adaptation as individuals create new applications would improve Artificial Intelligence tremendously. This would benefit everyone in the long run.

I have met with little success. Apparently, it seems that immediate corporate profit is more important than sharing technology with the public. Therefore, the Ennon Corporation stands alone in offering superior Artificial Intelligence programming directly to the home computer enthusiast.

Announcing AN-83: The "Thinking" Program

Believe me when I say AN-83 is the real thing. It is a true "thinking" program that receives an initial "knowledge base" from a data file read when AN-83 is started. Using inductive and deductive logical analysis, this amazing program deduces everything from that data and adds it to its memory. Conversing with you, AN-83 adds and combines with facts already known. It generates new conclusions not explicitly contained in its original knowledge base—just like your own thinking process! The result: it knows considerably more than the specific facts given to it.

AN-83 can also think about anything. It is virtually unlimited in its application. Think of your possibilities. The potential is limitless. In the right hands, AN-83 would revolutionize the adventure, strategy and other smart game-playing programs to say nothing of classic arcade games. On the other hand, AN-83 could be one of the most powerful business analysts available to the home computer.

FREE SOFTWARE

In addition, you will be receiving free, Eliza—the most amazing conversational A.I. program to date. Run this for your friends and jaws will drop with amazement. Eliza's responses are so human, it's uncanny. An entertaining program, Eliza will answer once and for all the question: What can your computer do?

How to Learn Artificial Intelligence

You can be creative. Experiment and modify to fit your personal use because AN-83 and Eliza both possess source code in basic, the most popular easy to use language for the micro. Their extensive, easy to understand commands walk you through the source code step by step. It's surprisingly simple. Even the beginner can understand the "How and Why of A.I."

A Fantastic Savings

The real profit to Ennon Corporation will be your participation in the future of Artificial Intelligence. Therefore, I am pleased to say nobody will miss this chance because they could not afford it. AN-83 is priced to cover just a fraction of its research and developmental costs.

The "Thinking" Program AN-83 is just $21.57. What's more, the astounding Eliza is yours absolutely free.

I guess it's obvious that I want you to participate in the future of Artificial Intelligence. Forgive my excitement and enthusiasm but I just know you are going to be very happy and impressed that such things could be done with your computer. You just won't believe it. Please take this opportunity now. Simply fill out your coupon below and mail today. Don't miss out. It's such a wonderful future of discovery and excitement that awaits you.

With very best of wishes,

Vincent Kurek

Vincent Kurek

☐ Please send me the "Thinking" Program AN-83 for only $21.57. In addition, I will receive absolutely FREE Eliza—the most impressive conversational Artificial Intelligence program to date.

Available in disk only.
For use with the Commodore 64.

Name ____________

Address ____________

City ____________

State ______ Zip Code ______

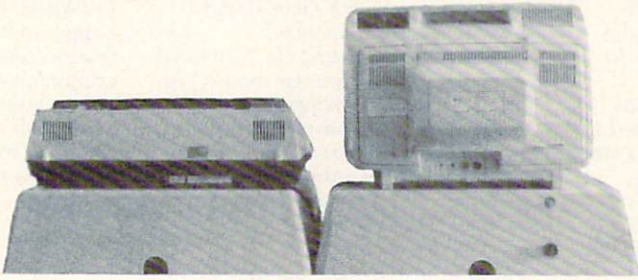
THE ENNON CORPORATION
1817 W. Call St., Suite B-8
Tallahassee, FL 32304

Canadian orders must be in U.S. dollars.
We export to other countries.
Commodore is a registered trademark.

INNOVATIVE ORGANIZERS, INC.

For Commodore 64™ & VIC-20™





WELCOME, to the family of **INNOVATIVE ORGANIZERS.** These **ATTRACTIVE** power organizer stands, made of **ABS PLASTICS, U.L. RATED,** provide an opportunity for a more **PROFESSIONAL** looking computer installation, both at **HOME** and in the **WORK PLACE.**

There is **AMPLE** room under the organizer to place all the computer accessories: **POWER SUPPLY, MODEM, GAME CARTRIDGES, MEMORY EXPANSION,** and **MUCH MORE.**

Now, let's examine how these organizers work. There are 3 CRT-Models.

CRT-1 holds a Monitor or T.V. Set and a VIC 20 or 64. All wires & cartridges fit neatly under the organizer.

CRT-2 same as CRT-1, but also has: **ONE-STEP MASTER SWITCH** (permits 4 pieces of hardware to be turned on and off from one switch, plus circuit protection, incoming power fused with 6AMP Fuse).

CRT-3 same as CRT-2, but also has: **SURGE & SPIKE PROTECTION** built in.

P-D-1: **HOLDS 1525 PRINTER & 1541 DISK DRIVE.** All wires & cartridges fit neatly under the organizer.

Ask about our Power Lock Switch & Power Portable Computer Fan & 2 Outlet Surge & Spike Protection.

CALL FOR LIST OF SOFTWARE AVAILABLE

IO. INNOVATIVE ORGANIZERS — YOUR ALTERNATIVE — INNOVATIVE ORGANIZERS

VIC 20 and Commodore 64 are trademarks of Commodore Electronics, L.T.D.

P.O. Box 108, Markleville, IN 46056

CRT-1	$17.98
CRT-2	$29.97
CRT-3	$37.97
P-D-1	$19.48

Optional Accessories

SS-1 Surge & Spike Suppressor $29.95

CF-1 Computer Fan $29.95

Plus Shipping

CALL TOLL FREE
1-800-233-3645 Ext. 15

Reader Service No. 150

SCUTTLEBUTT

COMBO DAISY WHEEL/DOT MATRIX PRINTER • ONE MEGABYTE DRIVE • VIC SPEECH SYNTHESIZER • FAST TAPE LOADER • GAMES FROM SSI, FIRST STAR, MASTERTRONIC • RUNNER'S LOG • COMPUTER MAILBOX • SCIENCE LAB SOFTWARE • STORYBOOK MAKERS • AUTO-REBOOTER

PASCAL ENHANCEMENT

Advanced Kyan Pascal, like the *Standard* version, is a full Jensen/Wirth implementation, but adding string handling, enhanced graphics, chain, include file, random access files, full pathname for files, and other extensions. Featured are a full-screen text editor and built-in assembler. The compiler, which generates optimized 6502 machine code, runs 8 times faster than *Standard Pascal* and 30 times faster than BASIC.

Advanced Pascal is available on a non-copy protected diskette for $69.95, *Standard Pascal* for $49.95.

Kyan Software, 1850 Union Street #183, San Francisco, CA 94123 (phone: 415-775-2923).

ESTIMATING SOFTWARE

Computerized Pricing Systems' *Estimating Software,* made available for over 25 micros since 1982, has been released in a 64, Plus/4, and 128 compatible version. If and when Commodore's LCD Portable is released, it will be made available for that as well. Price is $399.95; store owners may obtain a demo diskette for $19.95.

Computerized Pricing Systems, 3090 Oak Circle N., Broomfield, CO 80020 (phone: 303-469-0557).

OKIMATES

Okidata reports that the Okimate 10 color printer now supports 12 graphics packages: *DOODLE!, Koala Painter* (and Pad), the Edumate Light Pen software, *Super Sketch, Flexidraw, Paint Magic, Chalk Board, Sorcerer's Apprentice, Designer's Pencil, Color Me, Cadpak* and *Tech-Sketch.*

Okidata, 532 Fellowship Rd., Mt. Laurel, NJ 08054 (phone: 609-235-2600).

AUTOMATIC REBOOTER

Intended for bulletin boards, security systems, and other "constant use" applications for the C-64, the *ABL-64* cartridge will reboot and run an essential program after a power failure as soon as power is restored, even if the computer is left unattended. Price is $39.95.

Input Systems, Inc., 15600 Palmetto Lake Drive, Miami, FL 33157 (phone: 305-252-1550).



The Okimate 10 color printer is now compatible with 12 different graphics software packages. (See review on page 101 of the March '85* Ahoy!) *READER SERVICE NO. 125

NEW GAME RELEASES

Three for the C-64 from SSI:

Designed for the avid strategist, *Operation Market Garden* simulates Montgomery's 1944 paratroop raid on occupied Holland. As leader of the largest daylight airborne and surface operation in history, you coordinate airstrikes, engage in combat, blow up and rebuild bridges, and initiate cross-river assaults with airborne/engineer units. The battle is waged in 10 turns, from September 17-26. Price is $49.95.

Computer Ambush simulates man-to-man combat between your ten infantrymen and a (usually) hidden enemy. An onscreen map shows buildings, hedges, walls, doors, windows, and the location of your soldiers. Price: $59.95.

Newly adapted for the 64, *Cartels and Cutthroats* puts you in charge of a manufacturing plant. Based on newswires, market summaries, profit and loss statements, and memos from department heads, you must make executive decisions. Ulcer-inducers include inflation and high interest rates, labor demands, and government intervention. Price is $39.95.

Strategic Simulations Inc., 883 Stierlin Road, Bldg. A-200, Mountain View, CA 94043-1983 (phone: 415-964-1353.

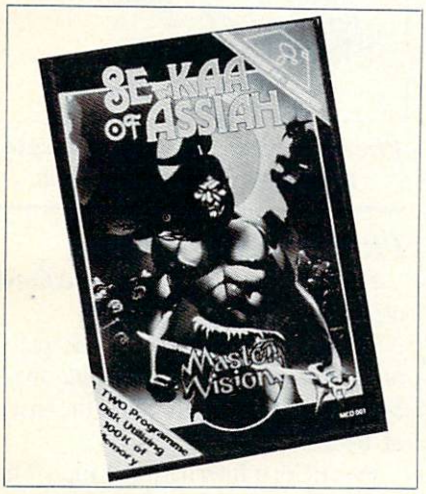
Encouraged by the success of their *Spy Vs. Spy* game, based on characters from *Mad Magazine,* First Star will release a line of *Super Powers*

programs beginning in September, starring comic book heroes like Superman and Wonder Woman.

Additionally, First Star announced that they will release *Boulder Dash* in a combination Commodore-Atari disk format.

First Star Software, 18 East 41st Street, New York, NY 10017 (phone: 212-532-4666).

Big Ben Games, Ltd. will import a number of British games for the 64 and VIC at prices of $16.95 (disk) or $14.95 (tape). None have ever before been sold in the United States. The titles of some of the releases tell you why: *Bizy Beezzzz, Choc a Bloc Charlie, Bumping Buggies, Son of Blagger, Cave Fighter,* and *Flight Path 737.* Coming is a text adventure based on the BBC program *Tripods.*

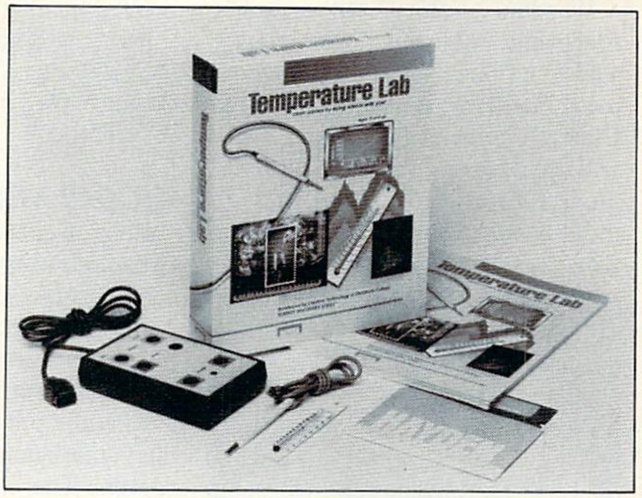


Two 50K games for $9.99, or 10¢ a K.
READER SERVICE NO. 127

For a free catalog, write Big Ben Games, Ltd., Box 875, Wilmette, IL 60091 or call 1-800-421-5300 (ask for operator R37).

British gamemaker Mastertronic has released *Se-Kaa of Assiah,* comprising two complete 50K programs. The gamer is required to find the Rod of Light, the Hammer of Vib-Ra, and the Casket of Vib-Ra and return them to their proper resting places, thus saving the world of the far future from the threat of Dark Hordes. Price is $9.99.

Mastertronic International, Inc., 407 Park Avenue South, Suite 16A, New York, NY 10016 (phone: 212-213-0166).



Temperature Lab *enables the user to choose between Fahrenheit and Celsius readings, to compare temperature scales on the same screen, and to display data in either graph or table form. READER SERVICE NO. 126*

EDUCATIONAL RELEASES

Temperature Lab is the first release in Hayden Software's science laboratory series, designed to transform a home or classroom into a science research center. The program, which enables students to conduct experiments that illustrate classic scientific phenomena, includes an electronic thermometer that plugs into the 64, enabling students to record temperatures and see them displayed on the screen. An Experimenter's Guide provides step-by-step instructions for experiments. Price: $99.95.

Soon to follow is *Light Lab,* focusing on various visible light concepts like foot-candles, solar eclipses, and sunsets.

Hayden Software Company, Inc., 600 Suffolk Street, Lowell, MA 01854 (phone: 617-937-0200).

CBS Software will adapt selected titles from its preschool product category for optional use with the Muppet Learning Keys Computer Keyboard (see review in March '85 *Ahoy!*).

CBS Software, One Fawcett Place, Greenwich, CT 06836 (phone: 203-622-2500).

Basic Math Competency Skill Building consists of eight tutorial and practice programs designed for math students from intermediate to high school. Each disk is $59.95; the 8-diskette series is available for $325.00.

Educational Activities, Inc., P.O. Box 392, Freeport, NY 11520 (phone: 516-223-4666).

Notebook Fun teaches students to keep neat notes. Tests, structured as games, are included. Price: $19.95.

AC3L Software, P.O. Box 246, New Derry, PA 15671.

STORY-WRITING SOFTWARE

Build a Book About You allows an adult to incorporate the name, address, pet, school, and friends of a child into a 32-page hardcover storybook. Available titles are *The Mystery of Scented Mountain, The Holiday Dragon, Adventures on the Riddle Planet,* and *The Greatest Circus Story Ever Told* (all written for children 2-12 years old). Each kit is $39.95; refill kits are $19.95.

Scarborough Systems, Inc., 55 S. Broadway, Tarrytown, NY 10591 (phone: 1-800-882-8222).

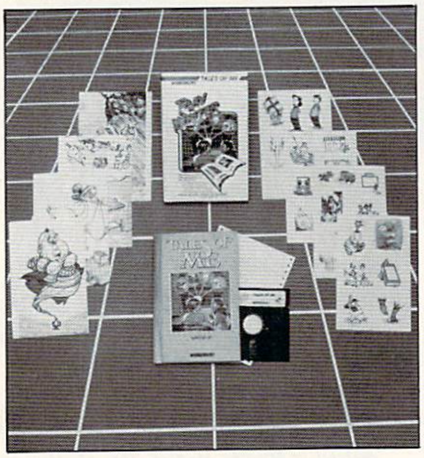
Two additions to Woodbury Software's *PlayWriter* lineup of home-makeable hardcover novels, designed for advanced young users and adults (the earlier releases *Tales of Me* and *Adventures in Space* were recommended for children aged 7-14):

Mystery! allows the writer to choose and describe his sleuth, determine method and motive of murder, and create his own cast of characters. If you want to be surprised, the program will pick the murderer for you.

Castles & Creatures lets you construct a fantasy world replete with dragons, knights, princes and princesses, and sorcerers.

All four *PlayWriter* titles sell for $39.95 and include disk, color stickers, and full-page illustrations.

Woodbury, along with Grolier Electronic Publishing, will sponsor a novel writing contest for students—using *PlayWriter* software, naturally. Children in the 4th grade and below will use *Adventures in Space,* 5th-6th grade *Castles & Creatures,* and 7th grade and up *Mystery!* Books will be judged on content, originality, gram-



PlayWriter ***line spans all age groups.***
READER SERVICE NO. 128

mar, spelling, and overall effect, with a $1,000 prize and a home computer for the winner in each category. Entries will be accepted from October 15, 1985 to January 15, 1986; entry blanks will be distributed through schools and retailers.

Woodbury Software, 127 White Oak Lane CN#1001, Old Bridge, NJ 08857 (phone: 201-679-0200).

SIGHT & SOUND ENHANCEMENTS

Sight & Sound has added bonus programs to their *Kawasaki Rhythm Rocker* and *Incredible Musical Keyboard* disks:

Rhythm Rocker now contains a new notation system that will display everything the user plays on the screen, a score printer, auto-correct, and the ability to change voices while playing, transpose to any of 12 keys, and use a multitude of specially created effects. Price remains at $34.95; current owners can upgrade for $15.00.

Incredible Musical Keyboard now allows for recording songs using up to three voices and sports a new notation and graphics program and five background accompaniments. Price is $39.95; upgrade is $7.00.

Sight & Sound Music Software, Inc., P.O. Box 27, Dept. R2D2, New Berlin, WI 53151 (phone: 414-784-5850).

VIC-TALKER

Long overshadowed by that other Commodore computer, the VIC 20 can now make some noise with VIC-Talker. Using advanced English pronunciation rules and a user-expandable exception memory, Talker will supposedly provide unlimited vocabulary translation of text to synthesized speech. The cartridge will operate immediately upon power-up with an unexpanded VIC 20. The user can vary pitch to synthesize a singing voice or emphasize statements, questions, and exclamations.

Priced at $89.00 plus shipping and handling, VIC-Talker will be shipped within four weeks of receipt of order by Talktronics, Inc., 27341 Eastridge Drive, El Toro, CA 92630 (phone: 714-768-4220).



Speech synthesizer for the VIC 20.
READER SERVICE NO. 129

RUNNER'S LOG

Jogger offers no training schedule for runners, but will enable C-64 users to compute weekly, monthly, and annual statistics based on regular entries of time, mileage, and weight. The information is available in the form of tabular data on screen or printer, or as 23 different hi-res graphic plots. Price is $29.00.

Parsons Software, 1920 Briar Meadow, Arlington, TX 76014 (phone: 817-465-4720).

FAST TAPE LOADER

In addition to allowing your datacassette to load tapes eight times faster (or 20% faster than your 1541 will load a disk), the *Express Tape Operating System* cartridge provides commands to recover lost or crashed BASIC programs. Available for $49.95 plus $3.00 postage (CA residents add 6% sales tax) from B.P.E., 22 Ethel Lane, Mill Valley, CA 94941 (phone: 415-388-0727).



Prices slashed! Hurry in and save!
READER SERVICE NO. 130

PRICE REDUCTIONS

Price reductions on assorted C-64 utilities:

PractiCorp has lowered the price of *PractiCalc 64* and *PractiFile* from $49.95 to $29.95 each, and the price of *64 Doctor* to $19.95.

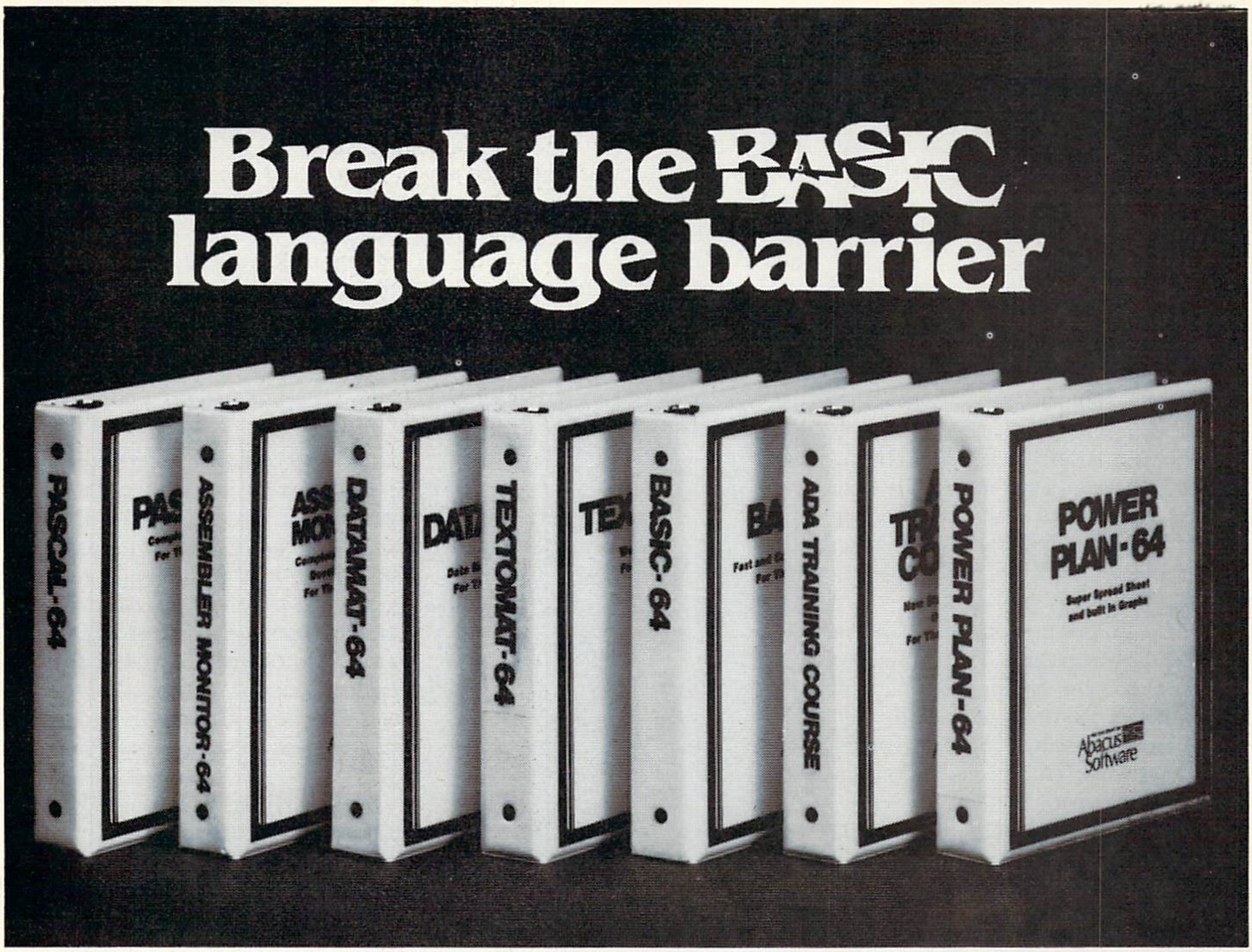
PractiCorp International Inc., The Silk Mill, 44 Oak Street, Newton Upper Falls, MA 02164 (phone: 617-965-9870).

International Tri Micro has lowered the price of *The Write File, Your Home Office,* and *Plus Graph* to $29.95 each.

International Tri Micro, 14072 Stratton Way, Santa Ana, CA 92705 (phone: 714-832-6707).

BETTER WORKING

The first title in Spinnaker's Better Working line of productivity software for the 64, announced several months back, has finally come available. *Spreadsheet* ($49.95) combines a full-featured spreadsheet with the basic text entry, editing, and word wrap capabilities of a word processor and the sort and search capabilities of a database. Slated for late summer release are *File & Report*



VIDEO BASIC-64 - ADD 50+ graphic and sound commands to your programs with this super development package. You can distribute free RUN-TIME version without paying royalties!
ISBN# 0-916439-26-7 $59.95

BASIC COMPILER 64 - compiles the complete BASIC language into either fast 6510 machine language and/or compact speedcode. Get your programs into high gear and protect them by compiling.
ISBN# 0-916439-17-8 $39.95

MASTER-64 - professional development package for serious applications. Indexed file system, full screen management, programmer's aid, BASIC extensions, 100 commands.
ISBN# 0-916439-21-6 $39.95

PASCAL-64 - full Pascal with extensions for graphics, sprites, file management, more. Compiles to 6510 machine code and can link to Assembler/Monitor routines.
ISBN# 0-916439-10-0 $39.95

ADA TRAINING COURSE - teaches you the language of the future. Comprehensive subset of the language, editor, syntax checker/compiler, assembler, disassembler, 120+ page guide.
ISBN# 0-916439-15-1 $59.95

FORTH-64 - loaded with hires graphics, complete synthesizer control, full screen editor, programming tools, assembler.
ISBN 0-916439-32-1 $39.95

C LANGUAGE COMPILER - a full C language compiler. Conforms to the Kernighan & Ritchie standard, but without bit fields. Package includes editor, compiler and linker.
ISBN# 0-916439-28-3 $79.95

ASSEMBLER MONITOR-64 - a macro assembler and extended monitor package. Assembler supports floating point constants. Monitor supports bank switching, quick trace, single step, more.
ISBN# 0-916439-11-9 $39.95

XREF-64 - indispensible tool for BASIC programmer cross-references all references to variable and line numbers.
ISBN# 0-916439-27-5 $17.95

OTHER TITLES ALSO AVAILABLE - WRITE OR CALL FOR A FREE COMPLETE CATALOG

Call today for the name and address of your nearest local dealer.

PHONE: (616) 241-5510

For postage and handling include $4.00 ($8.00 foreign) per order. Money order and checks in U.S. dollars only. Mastercard, VISA and American Express accepted. Michigan residents incl 4% sales tax.

FREE PEEKS & POKES WALL POSTER INCLUDED WITH EVERY SOFTWARE PURCHASE

You Can Count On

Abacus Software

P.O. Box 7211 Grand Rapids, MI 49510 - Telex 709-101 - Phone 616/241-5510

Reader Service No. 148

($49.95), a combined database and report generator; and *Word Processor,* featuring a spelling checker. All three programs will integrate.

Better Working, One Kendall Square, Cambridge, MA 02139 (phone: 617-494-1200).

BOOK RELEASES

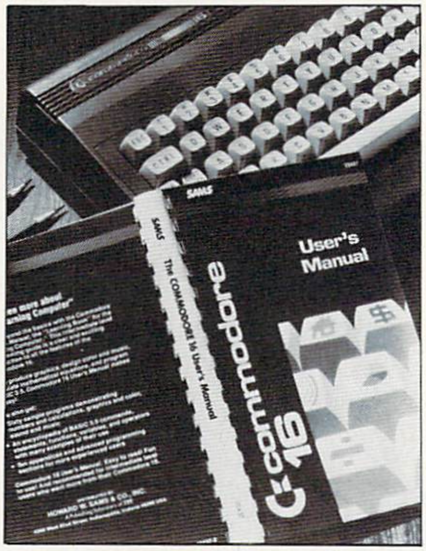
Two from Howard W. Sams & Co.: *The Commodore Plus/4 Book* ($16.95) compares the Plus/4 to other machines, provides an overview of its features, accessories, and software, and discusses its compatibility with C-64 peripherals. Also included are database and keyboard tutorials, an introduction to BASIC 3.5, and a chapter on system setup.

The Commodore 16 User's Manual ($12.95) helps new to intermediate users become familiar enough with BASIC 3.5 to work successfully with numbers, produce simple graphics, design color and music, and calculate mathematical equations.

Howard W. Sams & Co., Inc. 4300 W. 62nd St., Indianapolis, IN 46268 (phone: 317-298-5400).

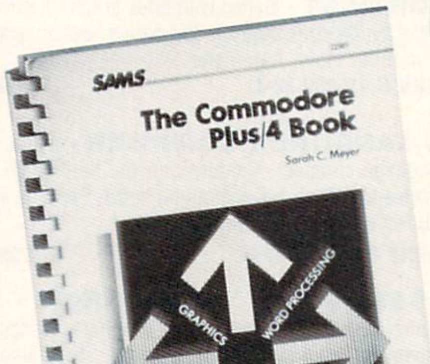
The World According to Robo the Robot ($12.95) introduces the reader to robotics, covering MEGO, the computer language of robots, the robot's brain, hardware chpis used, and a brief discussion of LISP, LOGO, and FORTH.

Hayden Book Company, 10 Mulholland Drive, Hasbrouck Heights, NJ 07604 (phone: 201-393-6306).



Covers BASIC 3.5, graphics, more.
READER SERVICE NO. 135

The Guidebook for Winning Adventurers ($9.95) provides answers to questions on various levels for such games as *Zork I-III, Enchanter,* and *Infidel,* plus information on how adventure games are programmed.



Software, features, accessories, etc.
READER SERVICE NO. 136

Baen Books, 8 West 36th St., New York, NY 10018 (phone: 212-947-8244).

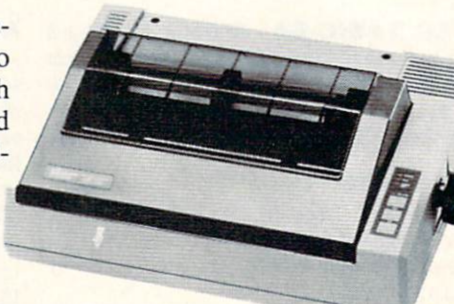
Sound and Graphics for the Commodore 64 ($14.95) takes the reader step by step through composing and playing melodies and sound effects, drawing cartoons, and then combining the sound and graphics.

John Wiley & Sons, Inc., 605 Third Ave., New York, NY 10158 (phone: 212-850-6000).

FINANCIAL PARTNER

Home users and businesses that write under 150 checks a month can use *Financial Partner* to keep track of income and expenses. The program will print checks, address labels and envelopes, and prepare and print a list of financial transactions, ledger accounts, income statements, and balance sheets. It will also prepare records for tax returns. Price is $74.95.

Practical Programs, Inc., P.O. Box 93104, 625 N. Milwaukee St., Milwaukee, WI 53203 (phone: 414-278-0829).



The SG-10C is Commodore-compatible.
READER SERVICE NO. 137

A STAR IS REBORN

Star Micronics' forthcoming SG-10C printer is an SG-10 (see review in May '85 *Ahoy!*) with a built-in Commodore interface, making it instantly compatible with the C-64, VIC 20, and C-128. Price will be $299.00.

Star Micronics, Inc., 200 Park Avenue, New York, NY 10166 (phone: 212-986-6770).

RELEASE THE POWER

OF YOUR COMMODORE



"Micro Mail" is the first dedicated and powerful mailing list designed exclusively for your Commodore 64. It's completely menu driven and loaded with prompts. Some say you don't need instructions, but we included them anyway. "Micro Mail" will add, delete, edit, search, display, print, and sort all your records in a matter of seconds. If you have a need to store and retrieve information, don't be without "Micro Mail."

"Micro Checkbook" is bound to be the standard for other programs to live up to. Like "Micro Mail," it is dedicated and very powerful. It's completely menu driven and very user friendly. "Micro Checkbook" will add, delete, edit, search, display, print, and sort all records in a matter of seconds as well as store all tax information and balance your statement every month. If your checkbook is a problem (and whose isn't?), "Micro Checkbook" is your permanent cure.

Available Soon: *Micro Calendar, Micro Inventory, Star Trivia and Post Time.*

⋆ ISLAND ⋆ DIGITAL ⋆ CONCEPTS ⋆

"Tomorrow's Standards, Today"

P.O. Box 1758
Staten Island, N.Y. 10314

 Reader Service No. 168

TELECOM NEWS

CompuServe subscribers who are members of Commodore user groups may now have unlimited access to the electronic edition of the Academic American Encyclopedia for $34.95 a year, or 30% off the regular price of $49.95 (connect charges extra).

Grolier Electronic Publishing, Inc., Dept. NH, 95 Madison Ave., New York, NY 10016 (phone: 212-696-9750).



Signalman Error-Free: detects errors.
READER SERVICE NO. 138

Two new products from Anchor Automation:

Designed for small businesses and individuals who want to utilize their microcomputers for mailing services, the Signalman Computer Mailbox



Signalman Mailbox: stores messages.
READER SERVICE NO. 139

($299) will receive and store incoming messages whether or not the user's computer is online, and provide remote access and message pickup with multilevel password protection. Memory is 64K.

Installed between modem and computer (or dumb terminal), the Signalman Error-Free ($199) detects and controls errors in asynchronous data communication.

Anchor Automation, Inc., 6913 Valjean Ave., Van Nuys, CA 91406 (phone: 818-997-7758).

CompuTrav, a directory of discount travel and vacations offered by a nationwide network of airlines, hotels, car rental agencies, and the like, offers discounts of up to 60% off regular prices. Charges are $1 per month plus 20¢ per minute connect time.

CompuTrav, 622 Broadway, New York, NY 10012.

The North American Online Directory 1985 lists 1511 machine-readable databases alphabetically, by 13 subject areas, and by 13 services offered. Price is $75.00.

R.R. Bowker Company, P.O. Box 1807, Ann Arbor, MI 48106 (phone: 1-800-521-8110).

Georgia OnLine, purportedly the nation's first regional information service, will provide Atlanta area users with information about the local community and services such as electronic mail and home shopping.

Georgia OnLine, Suite 720, Two Piedmont Center, Atlanta, GA 30305 (phone: 404-233-1915).

TWO INTERFACES

Compatible with all Commodore computers that use the Commodore type serial bus, the Printmaster/+G parallel interface ($119.95) allows full emulation of a 1525 or 801 printer, including full graphics and graphics characters. Advanced capabilities include the ability to display a disk directory without erasing a BASIC program, or a complete printer interface status. Available is a 16K buffer ($89.95) which provides faster graphics printing, renumbering and old utilities, and more features.

Omnitronix, P.O. Box 43, Mercer Island, WA 98040 (phone: 206-236-2983).

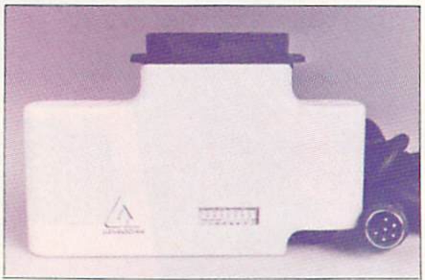
The Apricord CD ($69.95) will also allow a host of printers to produce all Commodore graphics characters. Utilizing a single-chip 65C02-compatible microprocessor that has



Printmaster/+G emulates 1525/801.
READER SERVICE NO. 140

a wider instruction set than the C-64 itself, it is reportedly the only parallel interface available that can be programmed in 65C02 code (a "superset" of C-64 assembly code).

Apricorn, 7050 Convoy Ct., San Diego, CA 92111 (phone: 619-569-9483).



Apricord CD: programmable in 65C02.
READER SERVICE NO. 141

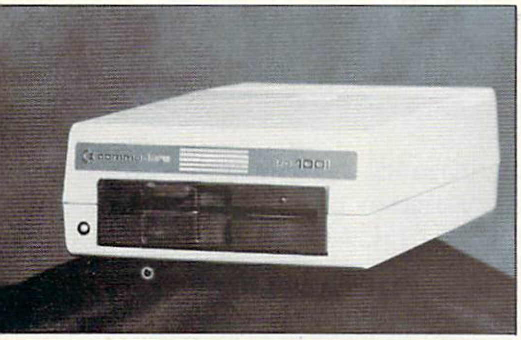
FORMULAS SOFTWARE

A program for calculating formulas commonly used in electronics (including Ohm's law, passive components, reactance, and others) is available for $10.00 from Jack Morrow, 7421 NW. 5th Terrace, Oklahoma City, OK 73127 (phone: 405-495-5071 before 1 p.m.).

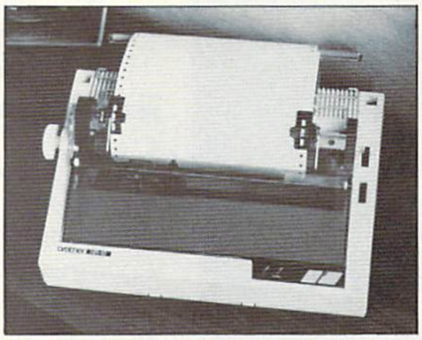
ONE MEGABYTE DRIVE

Commodore's SFD (Super Fast Drive) 1001 will have a one megabyte storage capability on a double-sided, double-density format. A utility disk with 64 and 8032 utilities will be included. The drive, contained in a case identical to the 1541's, will retail for $399.95.

Progressive Peripherals & Software, who will distribute the SFD for Commodore, will also manufacture



SFD-1001 can store 1M on disk.
READER SERVICE NO. 142



Brother HR-10: tractor, 2K buffer.
READER SERVICE NO. 143

Easy-Link, a serial to IEEE connector necessary to link the C-64 to the drive and other IEEE-488 devices. The unit, which will include its own power supply, will sell for $99.95.

Progressive Peripherals & Software, 2186 South Holly, Suite 200, Denver, CO 80222 (phone: 303-759-5713).

TWIN BROTHER

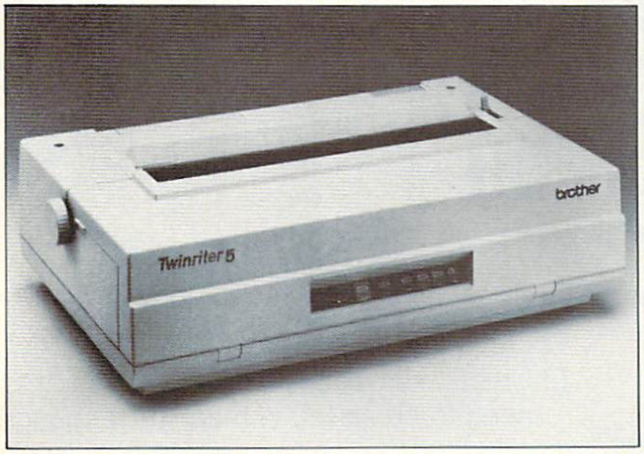
$300 is a lot to pay for a printhead. But that's what you're doing if you start out with a dot matrix printer and later add a cheap daisy wheel—because the printhead is the only real difference. It's always seemed ironic to us, in light of the miracle of the home computer, that no one can make a dual-purpose printer.

While the $1295 Twinriter 5 from Brother is intended for the small office environment, it *is* the first printer to feature combination daisy wheel and dot matrix printheads. Mode can be switched by pushbutton or software commands, even in the middle of a document.

Speed is 36 characters per second in daisy wheel mode, 140-160 cps using the nine pin dot matrix printhead. Included are a 3K buffer (expandable to 11 or 19K), and choice of six bit image densities.

Also new from Brother is the HR-10 daisy wheel printer with bidirectional printing, 2K buffer, and tractor feeder. Price is $349.

Brother International Corporation, 8 Corporate Place, Piscataway, NJ 08854 (phone: 201-981-0300).



Twinriter 5: combination dot matrix and daisy wheel.
READER SERVICE NO. 144

DUAL DISK CATALOGUER

Masterdual, like the previously released *Masterdisk,* is a disk cataloguer with disk repair utilities. The difference is that *Masterdual* is designed for 4040-format dual disk drives (such as the MSD SD-2).

The price of $29.95 includes a quick-reference translation card for equivalent hex, ASCII, binary, and decimal values.

Integrated-Software Systems, P.O. Box 1801, Ames, IA 50010 (phone: 515-233-2992).

25 PROGRAMS ON DISK

The Commodore 64 Programmer's Library includes over 25 utility, home management, and game programs on disk, with two additional disks supplying documentation and a utility for printing your own copies. Price is $25 postpaid in US and Canada; foreign orders $35 via international money order.

Baker Enterprises, 15 Windsor Drive, Atco, NJ 08004.

NEXT MONTH: ***Scuttlebutt*** **provides pre-release information on many Commodore-compatible products that will debut at June's Consumer Electronics Show.**

ERRATA

Because *DSKDU* (May '85) was written on a VIC 20, with its eight extra characters per line, line 2220 was too long to enter on the 64. C-64 users can amend this problem as follows:

```
2220 BY=P+L*4+C/2:HX=CAND1:H1$(0)=HX$(D%
(BY)/16):H1$(1)=HX$(D%(BY)AND15)
2221 C1=INT(C*1.5)
```

In the first installment of *Commodore Roots* (April '85), on page 52, it is stated that in hexadecimal, the letters A-F represent the decimal numbers 11-16 respectively. That should be 10-15. Thanks to Michael R. Whitner of Toledo, Ohio for alerting us to the error.

On page 35 of the May '85 issue, the phone number of Epyx is listed incorrectly. The correct number is 408-745-0700.

The *Font Factory,* described on page 7 of our June '85 issue, is available from Micro-W. Distributing, Inc., 1342B Rte. 23, Butler, NJ 07405 (phone: 201-838-9027).

We apologize for any inconvenience caused by the above errors. Remember that mistakes in *Ahoy!* programs and articles are posted on our bulletin board (718-383-8909) as soon as they are spotted.

Script Analysis

FOR THE C-64

By Bob Spirko

It's happened to all of us at one time or another. We see someone at a distance and although we cannot make out his features, we recognize him by his walk or his gestures. His body language discloses his personality. The hand, a part of the body, also reveals a person's character, but it has a singular difference – it leaves a tracing of its movement on paper. This tracing can be analyzed to determine personality, the science of which is called handwriting analysis, or graphology. A detailed analysis can take several hours to prepare, but a general analysis can be made simply and easily using *Script Analysis*. More important, you don't have to know a thing about handwriting analysis to use it.

The program will not analyze handwriting, but it will interpret your observations. Don't worry if you don't know what to look for. The program is rife with instructions and examples. I've made extensive use of character graphics to guide you along. All you do is answer the questions and the computer will do the rest.

While the analysis is general and less accurate than an indepth analysis done by a professional graphologist, it will provide you with a personality description of any sample of handwriting that you care to use. It was not feasible to incorporate the hundreds of characteristics that can be interpreted, so I chose a few dozen for their ease of recognition and general acceptance in graphology (opinions vary widely on some interpretations).

In the program, each handwriting characteristic is attributed words that describe personality traits. The program has a vocabulary of 74 words ranging from aesthetic to unreliable. The program compiles all the words or traits that appear in the script, but in the analysis it only lists those that appear more than once. Contradictions may turn up, but this does not necessarily mean the analysis is wrong. It is, for instance, possible to be both careful and careless. A person may be meticulous about writing a program while at the same time he allows his working space to degenerate into disarray!

The sample of handwriting should follow a few rules. For one thing, it must be written on a blank piece of paper; no guidelines may be used. Letter-sized paper is preferred. As for the implement, ball-point pen or pencil is best since pressure cannot be interpreted from a felt-tip pen. Probably the best samples to analyze are from letters, since they are usually written spontaneously.

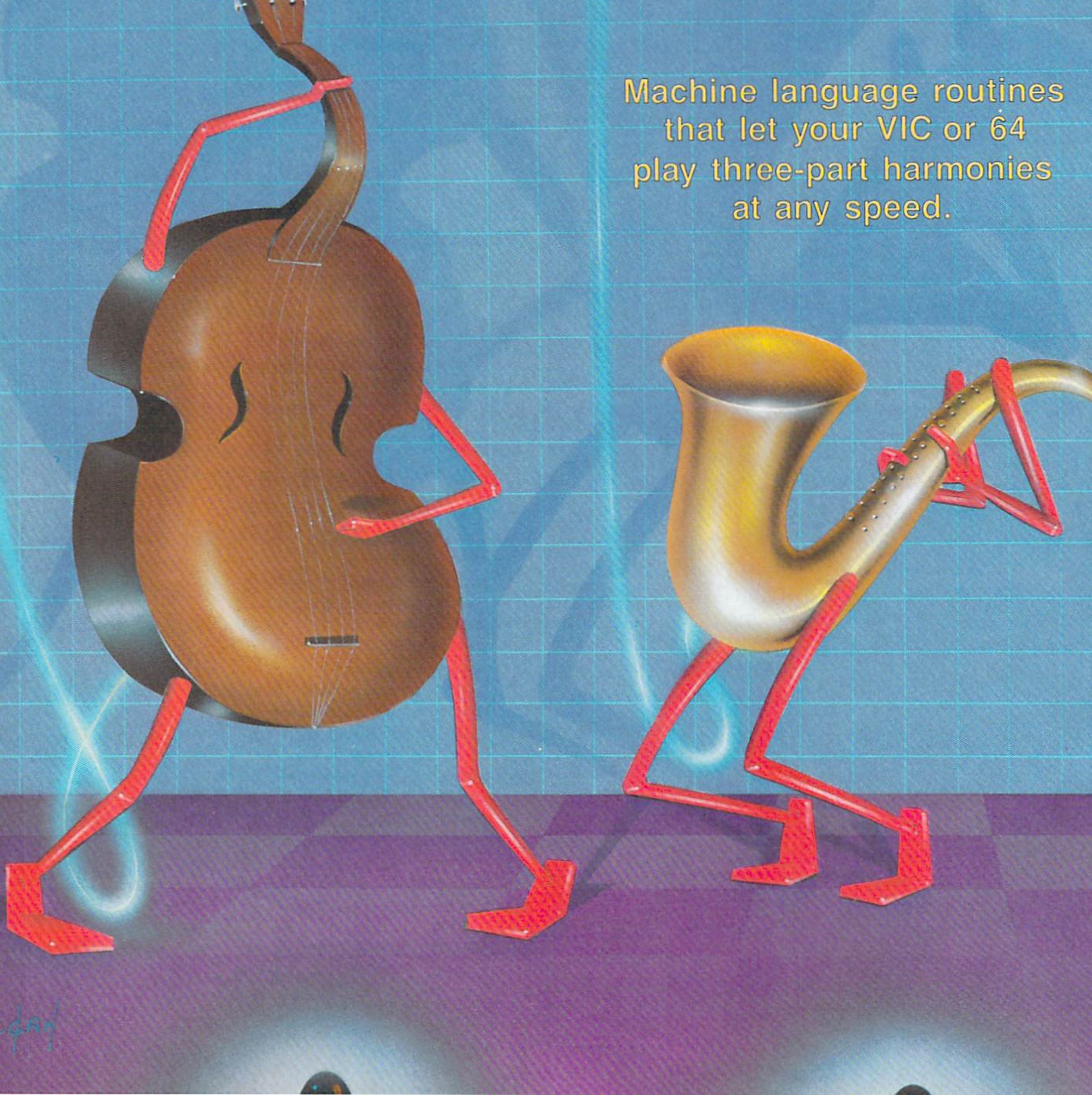
With your sample on hand, run the program. You'll first be asked for the filename. Next you'll be asked if it is a new file or if it is to be LOADed. If you press N you'll then be asked if the sample was written by a lefthanded or righthanded person. Answer this and you'll be given the first of fifteen categories, slant. Follow the instructions and then choose the appropriate handwriting characteristic. You'll go through each category this way. Once you've completed all of them, you'll be given a list of the handwriting characteristics that you chose, followed by two lists of personality traits. Dominant Traits are traits that have appeared three or more times, whereas Other Traits have occurred only twice. From there you have the option of sending the file to printer, disk, or tape; or you can elect to begin another analysis. □

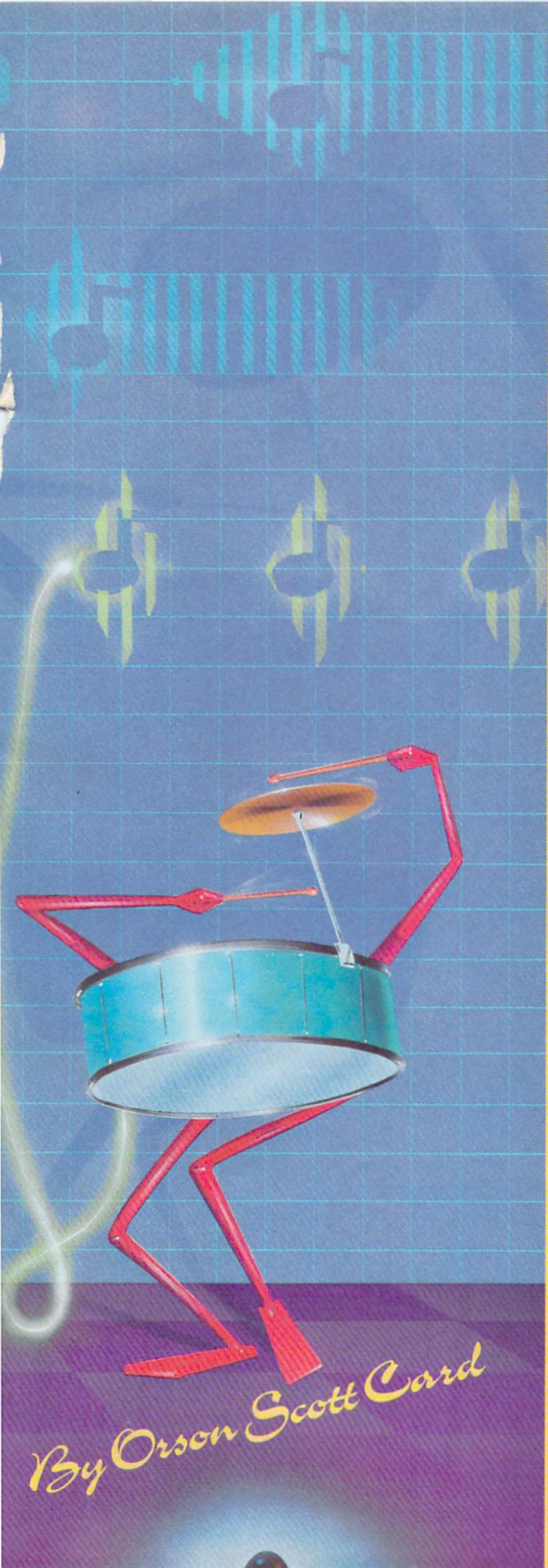
SEE PROGRAM LISTING ON PAGE 102

Trio Allegretto

FOR VIC AND 64

Machine language routines that let your VIC or 64 play three-part harmonies at any speed.





James Regan

CREATING YOUR OWN GAMES ON THE VIC AND 64

When you sit down at the piano, you don't have to be an expert musician. You hit a key and music comes out. If you hit too many keys, and you don't know what you're doing, you can turn it into noise pretty quickly. But the problem then is that you can make sounds too *easily;* you have to practice to get your own fingers under control.

That's because the piano is designed to do one thing and do it well. It translates the pressure of your fingers on the keys into strokes of a hammer against carefully tuned strings. All you have to do is figure out which keys to press.

"Oh, is that *all,* Mr. Card? Then you'll love to sit right down and play this little Chopin Nocturne. And when you've done that, I've got a Debussy, a Satie, and a charming little Mozart piece you'll be glad to diddle out for us."

No, I'm not saying it's easy. I'm just saying it's a heck of a lot easier than if you had to hold five or six hammers and hit the strings yourself!

Now, the VIC and Commodore 64 are not as conveniently designed as a piano. For one thing, the keyboard is designed with alphabetics, not harmonics, in mind. For another, the BASIC language completely ignores music in both computers–you have to use machine language or POKEs to produce the sounds. Even more limiting is the fact that you can never have more than three musical notes playing at the same time in either machine.

So you'll be delighted to know that this issue of *Ahoy!* includes machine language routines that you can embed in your own programs to play music that your program loads into BASIC strings. Mind you, unless you want to play nothing but the three songs I've included here as examples, you're going to have to do some calculating, and to make it work with your own program you'll have to do some programming, too. But these routines will allow you to play music as fast as you want to–too fast to hear, in fact–and using all three voices on either computer.

THE SHAPE OF THE SOUND

Even more important is the fact that both programs let you design the *shape* of the sound. This was fairly easy on the 64, since the SID chip was designed to allow sound "shaping;" it was much trickier on the VIC, and the results were far less effective, because the computer wasn't designed with sound shapes in mind.


EXPAND IT
NAVARONE
1
2
3
NOW FOR YOUR COMMODORE 64
EXPAND YOUR COMMODORE 64™* WITH NAVARONE'S NEW CARTRIDGE EXPANDER
Plug in up to three cartridges at a time. Select the program you want. Even reset the computer independently of the power switch.
• Reduces cartridge swapping
• Built in reset button
• Saves wear & tear
NOW ONLY $24.95
NAVARONE
NAVARONE INDUSTRIES, Inc.
19968 El Ray Lane
Sonora, California 95370
Phone: (209) 533-8349
Reader Service No. 172
*Commodore 64 is a trademark of Commodore Electronics, LTD. respectively. This product is manufactured by Navarone Industries for use on the Commodore 64 computer and is not made, sponsored, authorized or approved by Commodore Electronics, LTD.

Before you can effectively use these routines, though, you need to know something about sound.

The sound produced by a musical instrument does not have the same volume all the way through the note. For instance, a gong begins very loudly and then slowly fades away; a bowed violin begins much less sharply, but sustains its tone fairly loudly while it is still being played. A piano sound begins sharply, fades off rapidly to a lower sustain level, until the key or the pedal is released; then the note continues to "ring" until it fades to nothing.

The way that the volume varies during the sounding of the note is called the "sound envelope." The speed at which the sound rises from silence to the loudest volume is the "attack." The speed at which the sound then falls off after the attack is called the "decay." The volume that the sound stays at as long as the note is held is called the "sustain." And the speed at which the sound drops back to silence after the note stops being played is called the "release."

Notice that attack, decay, and release are all *speeds*—how fast the volume changes from soft to loud or loud to soft. Sustain is a volume control. The initial attack always takes the note to the maximum volume; the sustain is the fraction of that volume that the note has during the time it is held.

DOING ENVELOPES ON THE 64

A computer doesn't have to fuss with sound envelopes. You can simply switch on the note and it will continue to play that note until you switch it off. However, this makes for boring, mechanical sounding music.

So the designers of the Commodore 64 put the sound envelope entirely within your control. Each of the three musical voices has a separately controlled sound envelope, so that you can have the illusion of three different musical instruments playing at once.

Each of the three voices has four four-bit envelope control registers. Each four-bit register controls one of the envelope parameters—attack, decay, sustain, or release. Since four bits can hold any number from 0 to 15, the attack, decay, sustain, and release are all expressed as numbers from 0 to 15.

In the case of attack, decay, and release, the number represents speed: 0 is the fastest, and 15 the slowest. In the case of sustain, the number represents volume: 0 is silence, and 15 is maximum volume.

These four four-bit registers use up a total of 16 bits. Since they are read from ordinary eight-bit memory locations, those 16 bits are carried in only two memory locations for each voice. The first byte of each pair controls attack and decay; the second byte controls sustain and release. For voice 1, the registers are at locations 54277 and 54278; for voice 2, 54284 and 54285; for voice 3, 54291 and 54292.

If you POKE (or, in machine language, STA) these values into those locations, you have to change them a bit. Let's say you have a fairly sharp attack of 2, a gentle decay of 7, a low sustain of 5, and a slow release of 14,

and you want this envelope to be played by voice 1. First, you multiply the attack by 16 (16*2=32) and then add it or OR it (in ML, ORA it) with the decay value of 7. The resulting number, 39, is POKEd (or STAed) into location 54277. Then you multiply the sustain by 16 (16*5=80) and add it or OR it with the release value of 14. The resulting number, 94, is POKEd (or STAed) into location 54278.

Once that envelope is set for voice 1, every new note that you "gate" on—POKE 54276,(PEEK(54276)AND 254)OR 1—will use that same attack, decay, and sustain, and will keep playing the note at the sustain volume until you "gate" it off—POKE 54276, PEEK(54276)AND 254.

The 64 sound program, *Song Maker,* does all the calculating for you. All you have to do is set the individual attack, decay, sustain, and release values for each of the three voices in lines 9010, 9020, 9030, and 9040. AK%( ) is set to the attack for each voice; DY%( ) is set to the decay for each voice; and so on.

FAKING IT ON THE VIC

The VIC does *not* have separate envelope controls for the three sound generators, so the machine language routine has to fake the effect by changing the volume control at location 36879. Because all three voices are controlled by the same volume instruction, the three voices all have to have the same envelope. This does not mean, however, that they will *sound* the same. This is because each of the three voices has a different waveform, or tone quality *(see below).*

The *VIC Music* program has two sound routines. The BASIC statements from line 9010 to 9039 put the sound envelope routine in memory locations 663 to 767. The machine language commands contained in lines 9011, 9016, and 9025 contain subroutines that execute delays for varying lengths of time, ranging from milliseconds to the duration of an individual note.

The rest of the machine language routine executes as follows:

Line 9030: this subroutine is used constantly to set the volume register to the desired level. It reads the old value at location 36878, ANDs it with 240 to wipe out the old volume level, ORAs it with the current volume (which is maintained by another routine in location 150), and STAs it at 36878.

(If you aren't familiar with how this is done in machine language, here is a detailed explanation of each number in the routine:

173 is the command LDA (absolute), which means "pick up into the accumulator whatever number is found at the memory location described in the next two bytes."

14 is the low byte of the address of the memory location.

144 is the high byte, or page number, of the address. To find the decimal equivalent, you would use 144*156 +14, which is 36878. To go the other direction, from 36878 to low byte (LB), high byte (HB) form, you would use HB=INT(36878/256):LB=36878−(256*HB).

41 is the machine language command AND (immediate), which means "perform a bitwise AND operation between the number now in the accumulator and the number that immediately follows this instruction."

240 is the next number. It is ANDed with the number now in the accumulator, which we picked up a moment ago from location 36878. (This operation has the effect of zeroing out the volume-control bits, without disturbing the four bits at that location that have nothing to do with sound.) The result of this AND operation is put back in the accumulator.

5 is the command ORA (zero page), which means "perform a bitwise OR operation between the number now in the accumulator and the number found at the zero-page location described by the next byte."

150 is the address of the location where other routines store the number that they want used for the volume. Since the command 5 specified that this would be a zero- page address, the high byte of the address is assumed to be 0 and doesn't have to be stated. (This operation has the effect of combining the new volume value with the unchanged high byte of the original number at location 36878.) The result of this ORA operation is put back in the accumulator.

141 is the command STA (absolute), meaning "take the number in the accumulator and store it at the memory location specified by the next two bytes." It's the reverse of the LDA (absolute) command.

14 is the low byte again.

144 is, again, the high byte of address 36878.

96 is the command RTS, which means "return from this subroutine." The computer immediately jumps back to the next instruction after the JSR (address) instruction that invoked this subroutine in the first place.

For obvious reasons, we won't go into this kind of detail with the rest of the machine language routines included here.)

Line 9035: gradually bring the volume from 0 to 15, pausing between each increment of the volume for the amount of time specified by the attack value at location 659 (low byte 147, high byte 2).

Fly the unfriendly skies.



3D, scrolling flight simulation w. air-air & air-ground combat.

On-board computer, working gauges & 3 weapon systems.

3 skill levels, 15 scenarios, long-playing strategic depth.

You're in the cockpit of a dream machine—a bad dream for the poor sap whose tail you're waxing. You stick a silver bird up his exhaust and wince as he blossoms into fire. And then you rain down like pestilence upon the grunts in the tanks.

Skyfox. It's the fastest-selling EA game in history. It has the most awesome, high-speed animation you've ever seen on your computer. It's played to rave reviews in every magazine. And it's yours—at your retailer—or direct from EA at (800) 227-6703 with a 14-day satisfaction or money-back guarantee.

SKYFOX™

from ELECTRONIC ARTS™

ORDERING INFORMATION & MACHINE AVAILABILITY: Visit your retailer or call (800) 227-6703 (in CA call (800) 632-7979) for VISA or MasterCard orders. To purchase by mail, send check or money order to Electronic Arts, P.O. Box 306, Half Moon Bay, CA 94019. Add $3 for insured shipping & handling. Indicate machine verson desired. Allow 1-4 weeks for delivery. Apple II family available now at $39.95. C-64 available now at $34.95. For a copy of our complete catalogue and direct order form, send 50¢ and a stamped, self-addressed envelope to Electronic Arts, 2755 Campus Drive, San Mateo, CA 94403.

Reader Service No. 124

Line 9036: pause for a moment at peak volume.

Line 9037: gradually bring the volume down from 15 to the sustain value specified at location 661, pausing between each decrement of the volume for the amount of time specified by the decay value at location 660.

Line 9038: wait for the amount of time specified by the duration value stored at zero-page location 149 (another routine puts this number here), combined with the basic tempo set at location 170.

Line 9039: gradually bring the volume down from the sustain value to 0, pausing between each decrement of the volume for the amount of time specified by the release value at location 662. The volume is now zero, the note is finished, so return from this routine to the program that called it.

This is perhaps more detail than you wanted, but it does show you how a sound envelope is created. None of this is necessary in the 64, because the SID chip does all this automatically (and, I might add, much more smoothly and effectively); furthermore, the volume of all the notes is changed at once.

THE QUALITY OF THE TONE

The quality of a sustained note comes from the shape of the sound wave. The ringing tone of a gong comes from a very different sound wave than the blare of a trumpet, and a violin makes an altogether different tone. Note that this is not related to the sound envelope. A piano string, for instance, produces the same sound wave if it is plucked or if it is hammered, since the identical string is vibrating—but the sound envelope, the way the sound volume happens, is very different.

The 64 lets you choose one of three different musical waveforms for each voice. This waveform is selected by turning on bits at the control register (voice 1, location 54276; voice 2, 54283; voice 3, 54290). Remember that bit 0 of this location is the gate control—if bit 0 is a 1, the sound begins; if it is 0, the sound is released.

Bits 1 through 3 control sophisticated modifications of the tone quality that I won't attempt to explain here. (I've always found it somewhat harder to explain to other people things I don't understand myself.)

Bit 4 selects the *triangle* waveform. (Add 16 to the gate number.)

Bit 5 selects the *sawtooth* waveform. (Add 32 to the gate number.)

Bit 6 selects the *pulse* waveform. (Add 64 to the gate number.)

Bit 7 selects the *random noise* waveform. (Add 128 to the gate number. This is not used to produce melody or harmony.)

Now, I could spend a long time trying to tell you what the different waveforms sound like, but the truth is you'll only know what they sound like by trying them yourself.

In the 64 program, just change the values at line 9070 to select the waveform for each voice. You *can* combine the waveforms—that is, add both 32 and 64 or 16 and 32 or all three—but the result is less than thrilling.

However, you must choose at least *one* of the waveforms, or no sound will be produced. So the number you POKE in to gate the sound on will be *waveform* + 1; to gate the sound off, you can POKE a simple 0 or POKE in *waveform* + 0.

The pulse waveform is even more complicated than the others, because it can have different widths, which are controlled by the two memory locations just before the gate/waveform control register for each voice. The first of these bytes can use a value from 0 to 255; the second byte from 0 to 15.

WAVEFORM AND PITCH ON THE VIC

The VIC also allows you to choose different waveforms, but only by choosing the particular voice that has that waveform. When you choose to play voice 1, at 36874, you automatically get a low sawtooth waveform. Voice 2, at 36875, has a medium pulse waveform. Voice 3, at 36876, has a higher pulse waveform.

You use these registers to select the pitch—and to turn the voice on and off. The pitch is a value from 0 to 127. The pitches are exactly one octave different from voice to voice. That is, if you POKE the value 67 into register 36874, it will produce a C; if you POKE 67 into location 36875, it will also produce a C, but it will be an octave higher; a 67 stored in location 36876 will produce a C an octave higher still.

Whatever pitch you choose must be added to 128 to produce the actual sound, since 128 activates the on-off switch in each register. So to play that C in any of the three voices, you have to POKE *register, pitch* + 128.

The pitches are as follows:

7	67	97	112	C (B-sharp)
15	71	99	113	D-flat (C-sharp)
19	73	100		D
23	75	101		E-flat (D-sharp)
31	79	103		E (F-flat)
35	81	104		F (E-sharp)
39	84	105		G-flat (F-sharp)
47	87	107		G
51	89	108		A-flat (G-sharp)
55	91	109		A
59	93	110		B-flat (A-sharp)
63	95	111		B (C-flat)

To play each note, add it to 128 and POKE it into the register of your choice. To turn off the note, POKE a 0 into the same register.

HOW TO PUT MELODIES INTO THE 64

The program *Song Maker* uses exactly the same system of entering melodies in last month's program. To review briefly, the music is entered in strings starting at line 9500. The MD$( ) string contains the duration of each note of the phrase, expressed as a number from 0 to 9. (You can determine the actual length of each duration value at lines 9100-9105.)

The ME$( ) string contains the pitches for each note

in the phrase, expressed as alphabetic characters from A to G. These correspond exactly to the notes of the musical scale. To "sharp" a note, press SHIFT and the desired letter (C-sharp is SHIFT-C); to "flat" a note, press COMMODORE and the desired letter (B-flat is COMMODORE-B). To silence the voice (a musical rest or tacet), enter the @ symbol.

The MV$( ) string selects which octave the note should be played in. It is a value from 0 to 6. You only need to enter a number when the octave *changes;* as long as the melody is in the same octave, just enter a space. *All the strings for the same phrase must be exactly the same length.*

The program *Summertime* shows all the lines that need to be changed to enter a different melody. Once you have typed in and *saved Song Maker,* delete all the lines between 9500 and 9890 and then type in all the lines shown with *Summertime.* The program will now play a different melody with different waveforms and envelopes. With this song, all three voices play the same pitch every time, but with different waveforms and envelopes; the combination can be surprisingly effective.

The *Song Maker* program's chief value is that it translates standard musical notes to the numeric form the 64 can use. However, it is very time-consuming during the setup. The program provides three ways for you to get the pitch value to use directly in the program that you're creating.

First, the program will display each note's duration and pitch values on the screen, if you select the P option (press N to stop it from printing each note).

Second, the program will save the entire song on disk in the file you name in line 200. Third, the program will rewrite itself if you select the MAKE DATA option. Make sure, however, that you have already SAVEd a version of the program with your new melody on it, because the MAKE DATA option destroys all the lines in which you entered the melody using musical notes.

VITAL NOTE: no matter what else you do with this program, the variable F$ must be the very first variable assigned in your program. This is because the machine language routine takes the current note information directly from the very first variable in the variable table, which must be a string, and must contain the pitches of the notes for all three voices in order.

HOW TO PUT MELODIES INTO THE VIC

To use the program *VIC Music,* you can't use musical notation; you'll have to calculate the pitches yourself, using the table printed above. Your routine for entering the melody should be placed at line 9100. The variable SL must be set to the total number of notes in the melody. Then you must assign the duration of each note to the string MD$ and the pitches for each voice to the strings M1$, M2$, and M3$.

VITAL NOTE: it is absolutely essential that MD$, M1$, M2$, and M3$ be the first four variables mentioned in your program, *in that order,* because the machine language routine reads those string variables from the start of the variable table. Also, all four strings must contain exactly the same number of characters.

Notice that the melody included with this program assigns all the notes exactly the same duration. It could just as easily have read all the durations from a DATA statement, just as it now reads all the pitches. Also, notice that only M3$ is assigned its pitches from the DATA statements. M2$ and M1$ are then set to be the same as M3$. (Since M3$ has the highest voice, it will probably be most useful as the melody.)

Any pitch value from 0 to 127 will cause a note to be played, since it will be added to 128. If you want a note to be a rest or tacet, put it in the DATA statement as 128 or greater, since line 9110 converts any DATA value above 127 to 0, which shuts off the sound of that voice until the next note.

Half the fun, however, will come from your own experiments. Enter your own melodies and change the envelopes to see what happens.

(Note: If you program performs any cassette, disk, or RS232 operations after the sound routines are set up, they will have to be set up again, since the ML routines will be wiped out.)

That's enough for this month. Next month we'll finish with sound by playing around with sound effects for the VIC and 64. Then we can get back to something we can *see.* □ ***SEE PROGRAM LISTINGS ON PAGE 90***

PROM PROGRAMMING MADE EASY

A Guide to Permanent Program Storage in Read Only Memory

By Morton Kevelson

Most of us are well aware of the two types of memory inside our computers. We can hardly miss the significance of random-access memory (RAM), as the term is a prime selling point for most manufacturers. The C-64 has 64 kilobytes of the stuff, which is used for the temporary storage of programs and data. The VIC 20 comes with only five kilobytes of RAM, with provisions for adding up to 35 kilobytes externally. Actually, both the C-64 and the VIC 20 have an additional 1,024 nibbles of RAM (a nibble is four bits, or half a byte) which is used for the screen color map. In the C-64, these nibbles share the address space at $D8OO-$DBFF (55296-56319) with the character ROM and the system RAM.

Read-only memory (ROM) is not as well known to most users, since it plays a much smaller part in most promotional literature. Yet it is the programs in this permanent storage which give the computer its built-in personality. Both the VIC 20 and the C-64 are equipped with 20 kilobytes of ROM. The machine language programs which make up BASIC 2.0 and the operating system require 16 kilobytes of ROM. The remaining four kilobytes store the patterns for the Commodore character set. Without this permanent storage, the computer would be about as useful as a doorstop when first turned on.

Early users of microcomputers were not nearly as fortunate. In those bygone days of yesteryear (the early 70's) the personal computer had to be initially programmed by hand via a series of switches. This would usually allow for the loading of several hundred bytes of additional code, usually stored on a punched paper tape. This had to be done every time the computer was turned on. By the mid 70's things had improved somewhat. ROMs storing as much as 256 bytes had dropped in price to the point where a serious hobbyist could afford them. There was even talk of one kilobyte chips. As we shall shortly see, we have come a long way.

PROGRAMMING YOUR OWN

Several types of ROM are presently available. The kind that come in the VIC 20 and the C-64 are the mask programmable variety. These are used because of their extremely low cost (pennies per chip) in mass production quantities. For small users or hobbyists they are of little interest, as the initial setup costs are very high. The most popular type for individual users are the erasable programable read-only memories (EPROMs). These have always been considered essential tools by serious computer hobbyists.

The Commodore PET series of computers were even equipped with several empty chip sockets for the express purpose of installing additional ROM. In fact, these machines were not intended to have more than 32 kilobytes of RAM. Some quirks in the VIC 20 and C-64 operating system are relics from these forerunner machines. A notable example is the inability to perform a memory dump to tape from addresses above $7FFF (32767).

The designs of the VIC 20 and the C-64 have placed less emphasis on personalized PROM programming. This is in spite of the fact that both machines have provisions for considerable software on plug-in ROM cartridges. Nevertheless, supporting products for PROM programming have always been available for both computers. Those so inclined can readily avail themselves of this fascinating and useful technology. The state of the art is such that we feel at the very least every user group should consider acquiring the hardware for the convenience of their members.

Burning a ROM once referred to the process of placing a program into permanent memory. ROMs are no longer burned. This term is a holdover from the early days when some varieties of memory were programmed by literally burning open selected fusable links. This crude process has been replaced by the floating-gate avalanche-injection metal-oxide semiconductor transistor (FAMOS). These are specially fabricated field-effect transistors which

have the ability to store a minute electrical charge. In the discharged state the transistor looks like an open switch. This represents a binary one. When the charge has been placed, the switch is effectively closed to represent a binary zero. An EPROM chip is actually an organized conglomeration of these devices on a tiny bit of silicon.

HOW THEY WORK

A factory fresh PROM is completely discharged. In this state all the bits contain a one. If you were to examine an unprogrammed PROM you would find that each byte contains the value of $FF (256). The programming process consists of three steps. First, the proper address is selected. Second, the information to be stored is placed on the data lines. Finally, a high programming voltage (usually 24 volts) is placed on the programming pin. This voltage has to be very carefully controlled. If it is applied for too long, damage to the chip may result. If it is applied for not long enough, insufficient charge will be deposited and the stored information will be unstable.

EPROMs have one other characteristic which makes them so desirable as a development tool. Once the data has been properly programmed it is quite permanent, with a storage lifetime of many years. However, under the proper conditions, the data can be removed and the EPROM may be reused. In fact, with proper handling, there is no limitation on the number of times this cycle may be repeated. The EPROM is erased by illuminating the cells with ultraviolet light of a particular wavelength (2537 angstroms to be exact). A quartz window is thoughtfully provided on the chip package for this very purpose.

A PROM programmer is a specialized device designed to generate the necessary addresses, data, and programming voltages. A properly designed programmer will include enough built-in flexibility to handle a variety of EPROM types.

THE EPROM FLAVORS

EPROMs come in a variety of configurations. The most popular nowadays are the 2700 series, where the memory cells are arranged in an array which is eight bits or one byte wide (that is, addressing a single location returns a full byte of data). In contrast, the RAM chips in the C-64 are set up as a single bit per address. It takes eight chips in parallel to make up a single byte.

Memory chip capacities are specified in bits, not bytes. Thus, to obtain the chip's byte capacity, you have to divide by eight. The last two digits of the chip number represent the chip capacity in kilobits. For example, a 2764 contains 65,536 bits of storage capacity organized as eight kilobytes. Incidentally, the 2764 happens to be the most popular PROM chip for hobbyists as of this writing. It offers the lowest cost per bit among the available chip sizes. It also happens to be the capacity of the ROM chips used by the VIC 20, the C-64, and the 1541 disk drive for their operating systems. The 27128 is not far behind in a direct cost comparison. Its desirability hinges on placing 16 kilobytes of storage in a single socket. The 27256 (32 kilobytes) is also reasonably priced if you need the capacity. The 27512 (64 kilobyte) holds a whopping half a million bits of data. As of this writing, it is still priced beyond most hobbyists at several hundred dollars. However, we are all aware that prices in the semiconductor industry tend to drop rapidly as time goes on.

The user should also be aware of the number of pins required for the different chips. The ROMs in the VIC 20 and the C-64 are in a 24-pin package. All the chips mentioned above come in a 28-pin package. The manufacturers have made the pin layouts very similar, so that an adapter can be easily constructed. A 24-pin variety of the eight kilobyte EPROM is available. The model MCM68764 from Motorola fills the bill at a price which is currently five to six times as high as the 28-pin 2764.

When shopping for chips, keep in mind the other factors that affect price. These are primarily speed and power consumption. Given the slow clock speed of the C-64, the former is generally not a problem. Unless you plan on assembling some large memory arrays, the power consumption is not significant either.

ADDING MEMORY TO THE C-64

The Commodore 64 was specifically designed to allow for considerable memory expansion. This may seem a bit odd in light of the multiple memory layers which already fill parts of the computer's memory space. Nevertheless, every time a cartridge is plugged into the expansion port, memory is added to the computer.

For most game cartridges, this memory amounts to another eight or sixteen kilobytes of program on ROM. Generally, these cartridges are not true memory expansions. What actually occurs is a replacement of internal RAM and ROM by what is contained in the external cartridge. Most cartridges utilize only a small portion of the existing RAM. Thus a game cartridge can best be considered as turning the C-64 into a dedicated game machine while it is in place.

Some cartridges add to the C-64 memory resources. The Super Expander cartridge from Commodore contains eight kilobytes of ROM. When it's installed, the BASIC of the C-64 is enhanced by a set of graphics and sound commands. At the same time, eight kilobytes of BASIC RAM is hidden under the external ROM reducing available program space. The Super Expander cartridge does not waste this hidden RAM. It is utilized for the bit mapped graphics associated with the added BASIC commands.

Perhaps the most ambitious expansion of C-64 memory to date in a finished product is implemented by the Comal 2.0 cartridge. This contains the latest version of the Comal language on 64 kilobytes of ROM. As if this was not enough, an empty socket is provided on the board for an additional chip with a capacity for up to another 32 kilobytes.

The all-time prize for user expansion goes to the Jason-Ranheim Com-

pany, who make an eight-socket board which can accommodate up to 256 kilobytes of external ROM. Memory expansion need not be restricted to ROM. Additional RAM can be added to the expansion port. This is just the approach taken by the R. J. Brachman Co. with their CBUS II cartridge. We will be looking at both products later in this article.

HOW IT IS DONE

The designers of the Commodore 64 provided for all this expansion by building considerable flexibility into the cartridge port. To actually see how this is done, we will have to dig a little deeper. There are two primary devices which control the memory configuration of the C-64.

The first is a six-line I/O port built into the 6510 microprocessor. The internal hardware of the 6510 places this port at memory address 1. Its associated direction control register is located at address 0.

The second device is a custom designed integrated circuit, the Programable Logic Array (PLA). This chip is a collection of electronic switches (as are all logic devices) designed for the express task of managing the memory configuration requirements of the C-64. Internally it is responsible for the sharing of memory between the 6510 and the 6567 VIC II chip. Externally it is the control device for the expansion port.

The expansion port can best be described as a full microprocessor bus. It brings to the outside world the entire set of eight data lines and sixteen address lines of the 6510. In addition, there are several control lines which are associated with the PLA. Two of these lines, $\overline{\text{EXROM}}$ (pin 9) and $\overline{\text{GAME}}$ (pin 8), are hardware inputs to the PLA. These are normally held to a logical one represented by +5 volts in the C-64. The horizontal line over their labels indicates that their intented logical function is implemented when they are set to a logical zero or 0 volts in the C-64.

When a cartridge is plugged into the expansion port, one or both of these lines are grounded. This causes some of the internal memory of the C-64 to be replaced by whatever is in the cartridge. The affected memory areas are the eight kilobytes of RAM from $8000 to $9FFF (32768 to 40959). The eight kilobytes of BASIC ROM from $A000 to $BFFF (40960 to 49151) and the eight kilobytes of KERNAL ROM from $E000 to $FFFF (57344 to 65535). These lines work in conjunction with two output lines at the port, ROML (pin 11) and ROMH (pin B). The former addresses in the eight kilobyte $8000 block, while the latter controls either the $A000 or the $E000 block.

The effect of all four of these lines can be further modified by the 6510 I/O control port at address 1. This can be used to switch between ROM (either internal or external) and the built-in RAM that occupies the same address space. Some cartridges exert direct control of the hardware memory configuration by placing electronic switches at the EXROM and GAME lines. These "soft switches" are accessed by selecting a particular address for their control. Additional RAM or ROM can be switched in or out, usually in eight kilobyte banks, by installing additional control logic on the cartridge. This is the technique used by the Comal cartridge mentioned above.

SOME PROM APPLICATIONS

Most of us are familiar with the dozens of games and applications programs available in cartridge forms. The C-64 and VIC 20 offer other possibilities for custom ROMs. These range from changing the character set to direct modifications to BASIC and the operating system. The default language could even be changed. How about a C-64 that understands FORTH or PASCAL? Think of all the possible applications for dedicated autostart programs. This would allow the computer to be used as an unattended control device without the need for a disk drive or cassette deck. The list is endless. The three reviews that appear on the succeeding pages present some interesting products which offer some unique ideas of their own. ☐

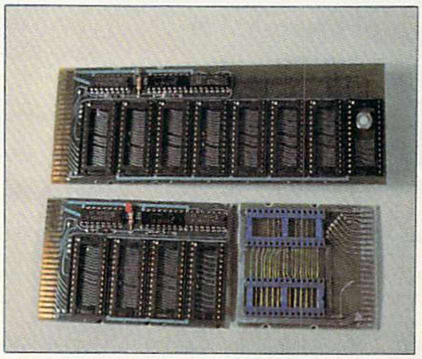
THE PROMENADE PROM DEVELOPMENT SYSTEM

Jason-Ranheim
580 Parrott Street
San Jose, CA 95112
Phone: 800-421-7731
(in CA 800-421-7748)

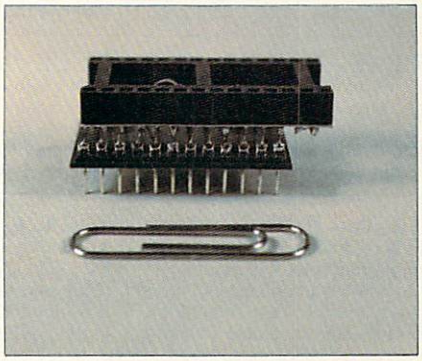
According to Webster's New World Dictionary, to "promenade" is to "go for a walk." According to the Jason-



The Promenade inside and out. The 4 x 5" package installs in user port.



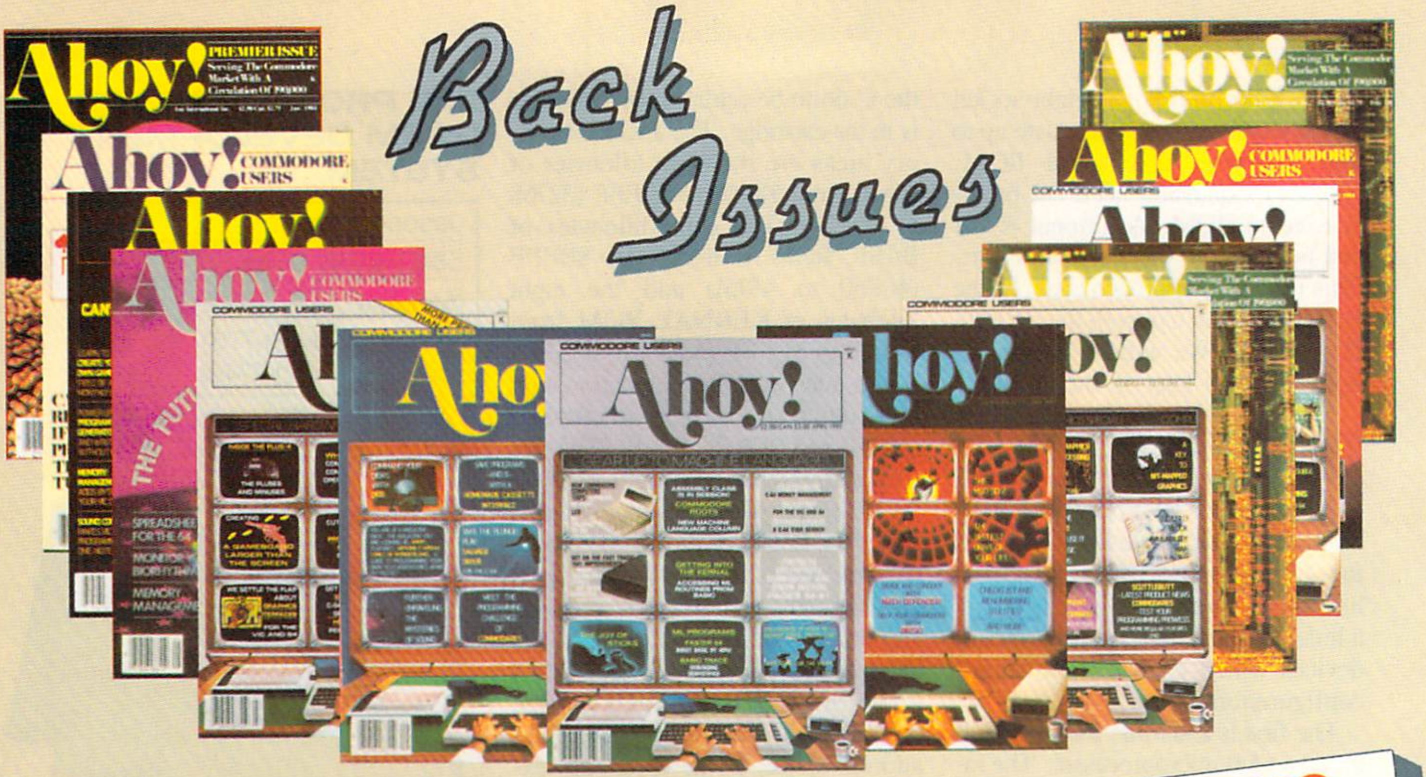
Top: the PCC-8 board; bottom left: the PCC-4; bottom right: the PCC-2.



Adaptor that converts a 28-pin 2764 to 24-pin format compatible with 64 & VIC.

Ranheim Company, the Promenade model C1 is the heart of a ROM pro-

Back Issues



ISSUE #1–JAN. '84 $4.00 64 v. the Peanut! Telecommunications! Tales of the Commodore! And ready to enter: Multi Draw 64 Graphics System! Interrupt Music Maker/Editor! A Peek at Memory! Sequential Files!	**Sorry...Issues #1, #2, and #7 sold out! Reprint editions (programs and feature articles only) available for $4.00 each.**		
ISSUE #2–FEB. '84 $4.00 Illustrated tour of the 1541! Artificial intelligence! Synapse's Ihor Wolosenko interviewed! And ready to enter: Music Maker Part II! Screen Manipulation! Night Attack! Relative Files!	**ISSUE #7–JULY '84 $4.00** MSD dual disk drive! Database buyer's guide! Training your cursor! Screen displays! File Sleuth! Users Groups! And ready to enter: Renumbering! Checklist! Math Defender! Brisk!		
ISSUE #3–MAR. '84 $4.00 Anatomy of the 64! Printer interfacing for VIC & 64! Educational software series begins! And ready to enter: Address Book! Space Lanes! Random Files on the 64! Dynamic Power!	**ISSUE #8–AUG. '84 $4.00** Choosing a word processor! Computational wizardry! Creating your own word games! Sound on the 64! And ready to enter: Micro-Minder! Directory Assistance! The Terrible Twins!	**ISSUE #12–DEC. '84 $4.00** Buyer's guide to printers! 1525 printer tutorial! Custom characters! User Guide to KMMM Pascal! Diving into BASIC! And ready to enter: Construction Co.! Space Patrol! Cross Ref!	
ISSUE #4–APR. '84 $4.00 Petspeed and Easy Script tutorials! Printer interfacing continues! Laserdisc! And ready to enter: Apple Pie! Lunar Lander! Name that Star! Lower case descenders on the 1525 printer!	**ISSUE #9–SEPT. '84 $4.00** Program your own text adventure! Build a C-64 cassette interface! Video RAM! And ready to enter: Salvage Diver! DOS! Sound Explorer! The Castle of Darkness! Base Conversions!	**ISSUE #13–JAN. '85 $4.00** VIC/64 OS exposed! Sprites! 1541 device # disconnect switch! Ghostbusters! And ready to enter: Ultra Mail! Music Tutor! Alice in Adventureland! Midprint! To the Top! Tape/Disk Transfer!	**ISSUE #16–APR. '85 $4.00** Assembly language column begins! Programming the joystick! 1541 disk drive alternatives! And ready to enter: Hop Around! Faster 64! Booter! Elecheck! BASIC Trace! Space Hunt!
ISSUE #5–MAY '84 $4.00 Future of Commodore! Inside BASIC storage! Memory management on the VIC & 64! Guide to spreadsheets! And ready to enter: Math Master! Air Assault! Biorhythms! VIC Calculator!	**ISSUE #10–OCT. '84 $4.00** C-64 graphics programs! Bit-mapped graphics! Joystick programming! And ready to enter: VIC 40 Column Operating System! BAM Read & Print! Emerald Elephant! Lawn Job!	**ISSUE #14–FEB. '85 $4.00** Printer interfacing! Multicolor sprites! Modems! Bulletin boards! Theory of game design! And ready to enter: Futurewar! Fontasia! VIC Eraser! Insurance Agent! Flankspeed! Telelink 64!	**ISSUE #17–MAY '85 $4.00** Disk drive enhancements! Install a reset switch! Assembler escapades! And ready to enter: Super Duper! Two-Column Directory! DSKDU! Raid! DOS Plus! Font Editor! Tile Time!
ISSUE #6–JUNE '84 $4.00 Game programming column begins! Program generators! Rupert on inputting! Memory management continues! And ready to enter: Post Time for the 64 & VIC! Alpiner! Sound Concept!	**ISSUE #11–NOV. '84 $4.00** Music programs & keyboards for the 64! Graphics feature continues! And ready to enter: PTE word processor! Block Editor! Alternate Character Set for the 64! The Tunnel of Tomachon!	**ISSUE #15–MAR. '85 $4.00** Making multiscreen gameboards! Inside the Plus/4! Commodore DOS! And ready to enter: Old Routine! Programmable Functions! Automatic Line Nos.! Home Budget! Salmon Run!	**ISSUE #18–JUNE '85 $4.00** Music & graphics entry systems! How modems work! Inside the 6510! And ready to enter: Quad-print! Mapping 4.4! Towers of Hanoi! Speedy! Duck Shoot! The 6510 Simulator!

STOP PROGRAMMING...

...until you have a complete collection of *Ahoy!* behind you! Our articles and tutorials can save you hours of toil at your Commodore keyboard...our programs, hundreds of dollars at the software counter! Use the form below to order your issues while limited supplies last!

Use coupon or facsimile. If ordering more than three issues, list choices on separate sheet.

If you have a modem and want more complete information on any of the back issues listed, call *Ahoy!*'s Bulletin Board Service at 718-383-8909.

Ahoy! BACK ISSUES
ION INTERNATIONAL INC.
45 West 34th Street
Suite 407
New York, NY 10001

Please Send Me The Following:

_____ Copies of issue number _____
_____ Copies of issue number _____
_____ Copies of issue number _____

Enclosed Please Find My Check or Money Order for $_____
(Outside the USA please add $1.00 for every copy)

NAME_____
ADDRESS_____
CITY_____ STATE_____ ZIP CODE_____

gramming system for use with the Commodore 64 or the VIC 20. Stand-alone ROM programmers are relatively expensive items. Much of the expense resides in supplementary hardware requirements. These include memory for data and control program storage, a controlling microprocessor, and of course a power supply. The Promenade utilizes the existing facilities of the computer to create a very effective ROM programming environment. The capabilities of a Promenade with a C-64 or a VIC 20 and a 1541 disk drive actually exceed those of more expensive dedicated units. That includes the cost of the computer and disk drive.

VIC 20 users take note. The Promenade system will function properly without any memory expansion. However, programming ROMs which exceed the memory capacity of the unexpanded VIC 20 will be somewhat inconvenient. A minimum of eight kilobytes of expansion RAM is recommended.

THE HARDWARE

The Promenade itself is a slim 4½ x 5" package in a brushed aluminum case. In use, it is installed in the computer's user port. The top surface of the Promenade is dominated by the 28-pin zero insertion force (ZIF) dual in line package (DIP) socket flanked by green, red, and yellow indicating lights. These light emitting diodes serve to indicate the status of the system at all times. The first (green) simply displays the presence of power to the Promenade. The second (red) indicates when the ZIF socket is energized, such as during a ROM read operation. It also serves as a warning not to insert or remove a PROM when it is lit. The third (yellow) alerts the user to a ROM programming activity. It also flashes when some kind of programming error takes place.

THE OPERATING SYSTEM

The *PROMOS* software, supplied on either cassette or disk, makes the reading and programming of PROMs a snap. *PROMOS* is designed to let the user work with a PROM in a fashion which is very similar to using the cassette or disk drive. In fact, *PROMOS* assigns device number 16 to the Promenade. When activated, *PROMOS* behaves very much like a DOS wedge. It makes full use of the computer's operating system. Programs can be LOADed and SAVEd. Files can be OPENed and CLOSEd. Data can be written with the PRINT# and CMD commands and retrieved via the GET# and INPUT# commands. There is even a directory command, the dollar sign ($), which returns the contents of the ROM.

PROMOS reserves part of the ROM space for the directory information. Unlike the disk drive, the *PROMOS* directory is spread out through the ROM with the associated files. A series of links, similar to BASIC's internal line links, keeps track of each file's relative position. Complete details of the *PROMOS* file structure are included in the twenty-page manual.

The *PROMOS* file-oriented operation requires the presence of the Promenade to function. This limits the capacity of the PROM filer to what can be stored in a single chip. The Promenade can handle chips with a capacity of up to 512 kilobits (64 kilobytes). As mentioned earlier, chips with this capacity are still quite expensive. Thus most users will limit themselves to eight or sixteen kilobytes of online storage. A useful enhancement to *PROMOS* would extend this file-structured operation to the PCC-2, -4, and -8 expansion boards discussed below. This enhancement would allow for the creation of a ROM disk with up to a quarter megabyte capacity.

PROMOS also includes two direct memory commands, the Greek letter pi and the British pound symbol. These allow for the direct bidirectional transfer of data between the computer's memory and the PROM. The command syntax is very similar to that used by most machine language monitors. *PROMOS* has been designed to be highly compatible with most machine language monitors. The commands will generally work from within a monitor, although an occasional question mark indicating a monitor syntax error does appear. The solution is usually just to reissue the command. The manual recommends *Hesmon* for use with *PROMOS*. A cartridge-based version which includes *PROMOS* and a DOS wedge is available for $24.95. We found that *PROMOS* seemed to work well with *Micromon*, a very powerful public domain machine language monitor.

The Promenade has enough built-in flexibility to handle a large variety of EPROMs of varying capacities and types. *PROMOS* accepts two parameters which allow the system to distinguish among the various types of EPROMs. The "control word" number tells the operating system which type of EPROM is being used. The manual lists 23 PROM types, to which the 27512 has recently been added. This includes the 48016P, which is an electrically erasable device (EEPROM) for which *PROMOS* includes a special erase command.

The second parameter lets *PROMOS* optimize the programming operation. The Program Method Word (PMW) selects from four methods of PROM programming. One is based on the chip manufacturers' recommendations. The other three are "intelligent" methods which optimize programming time. The method developed by Jason-Ranheim uses a series of increasingly longer pulses till the data has been verified. Thus for an eight kilobyte ROM, which would normally take seven minutes using the standard method, we found programming times as short as 30 seconds with the Jason-Ranheim method.

PROMOS was kept as brief as possible to minimize its memory requirements. The code is fully relocatable by simply changing the top-of-memory vector before RUNning it. This brevity does leave the program open to operating problems. Under some conditions it is possible to crash the program while all of the Promenade lights are still on. If a PROM is in the socket, damage to the chip might result. A hardware protect switch to disable the socket could be a useful addition.

OTHER SUPPORT PRODUCTS

Jason-Ranheim offers several circuit boards for installation in the computer expansion port. A VIC 20 board (PVC-2) and a C-64 board (PCC-2) priced at $6.95 each support up to two eight-kilobyte ROMs. For the VIC 20, any one of the four eight-kilobyte expansion blocks can be used. The C-64 board uses the two eight-kilobyte blocks located at $8000 to $BFFF. Instructions are provided on how to store a BASIC program in ROM which will autostart when the computer is turned on. Two program listings are provided, one which RUNs the program from cartridge memory and the other which downloads the program to the usual BASIC memory space before RUNning.

Also available are two additional bank switching circuit boards. The PCC-4 ($19.95) will accommodate up to four ROMs with either an eight-, sixteen-, or thirty-two kilobyte capacity for a total of 128 kilobytes of add-on memory. This memory will appear in eight-kilobyte bank switched chunks at the $8000 block. The bank select register can be located at either $DFFF (57343) or $DEFF (57087). Any eight-kilobyte bank or none at all can be switched in or out. The PCC-8 ($29.95) is similar, except eight sockets are provided for up to 256 kilobytes of external ROM. Complete instructions are included, along with a *DOWNLOAD/RUN* program for installing up to 38 kilobytes of program with an autoboot routine in ROM.

Before you go running off with visions of putting all your favorite applications in nearly instantaneously bootable form, keep several things in mind. Most commercial software is copy protected and in machine code. It will require a fairly knowledgeable user and some time to extract the programs and put them into the proper format for use with this method. So if you have the programming capability this presents an excellent way to get your favorite word processor, spreadsheet, and database all online at one time.

No PROM programming system would be complete without some means of erasing a previously programmed PROM. Jason-Ranheim does not make an ultraviolet eraser of their own, but they will supply "hobbyist" and "industrial" versions of these devices for $34.95 and $79.95 respectively.

CONCLUSIONS

The Promenade is a good general purpose PROM programming system. Some unique support products make it very attractive to the moderately advanced user. The built-in hardware and software flexibility, coupled with some excellent technical support, make it a good choice for turning the C-64 or VIC 20 into a PROM development system.

The Promenade is available for $99.95 with the supporting software on tape or disk. ☐

ADDENDUM

Jason-Ranheim is continuously coming up with new uses for the Promenade. As we went to press we received word of a new four-socket expander board (PRB-4) similar to the PCC-4. The PRB-4 is designed to accommodate two eight-kilobyte PROMs and two eight-kilobyte RAM chips. An onboard battery backup allows the RAM to retain data after the C-64 has been turned off. According to John Ranheim, president of the company, the PRB-4 is being used as an aid for educating the homebound handicapped. An examination is programmed into the PROMs. The student's answers are stored in the RAM. The battery backup retains the data for subsequent grading under computer control. Price is $24.95.

CBUS I & II:
THE CARTRIDGE BACKUP SYSTEM

R.J. Brachman Associates
P.O. Box 1077
Havertown, PA 19083
Orders 1-800-CBUS-C64
info 215-622-5495

The CBUS (Cartridge BackUp System) is a product which will undoubtedly evoke considerable controversy. It is marketed with the primary intent of allowing a user to copy and use the software which is stored in the ROM chips of a Commodore 64 program cartridge. This attitude, while probably reflective of good marketing sense, is somewhat unfortunate. As we shall see, the CBUS package does have a valid purpose in the C-64 scheme of things.

THE HARDWARE

CBUS is primarily a hardware accessory. The total system consists of two parts available separately or as a package.

The CBUS I is essentially a single slot expansion port extender board. This in itself is unremarkable. What makes the CBUS I unique is the inclusion of an eight position DIP switch. There is also a reset push button which does come in handy at times (see the May '85 *Ahoy!* for more on reset buttons). The DIP switches intercept the four cartridge control lines which were described in the introduction to these reviews. By properly manipulating these switches, the particular configuration of nearly any 16 kilobyte cartridge can be determined. Once this is known, the cartridge contents can be downloaded to disk. This can easily be done with a machine language monitor or with the custom program listed in the manual.

Once on disk the cartridge can be dispensed with. The program can be then loaded into the C-64's resident RAM and RUN from there—well, almost. As it turns out, very few cartridge-based programs are directly amenable to this sort of treatment. For various reasons, most cartridge software will require some modifications to RUN in this fashion. Since these programs are nearly always written in machine language, this task is not for the uninitiated. A select number of cartridges cannot be so converted, as they utilize the underlying RAM which shares their program space.

This is where CBUS II comes into the picture. Although it physically resembles the typical cartridge which it is intended to replace, internally

there is a world of difference. In place of the expected ROM, the CBUS II contains sixteen kilobytes of static RAM. To complete the picture, the CBUS II contains a hardware "personality latch" which allows the RAM to emulate any of seven possible 16-kilobyte cartridge configurations which are available for the C-64.

To complete the picture, a program listing is included with the documentation which will LOAD in a cartridge "snapshot" and configure the personality latch accordingly. Once loaded, the cartridge program can be started by locking the latch with the RUN/SET switch and then hitting the cartridge reset button.

THE DOCUMENTATION

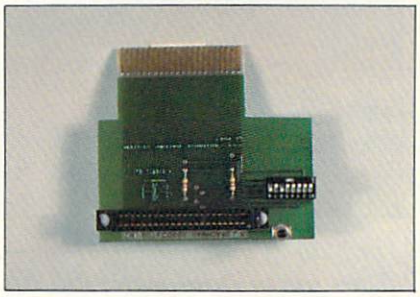
Each of the CBUS modules come with its own instruction manual. The CBUS I manual provides a thorough description of the various cartridge configurations. Included are complete instructions on how to identify a cartridge with the CBUS I. Two brief programs are listed which allow you to create a menu-driven disk-loaded version of all your cartridges. The manual also includes a detailed breakdown of all the possible C-64 memory maps and how they are created. This is one of the most comprehensive discussions we have come across on the subject. The interaction between the 6510 internal control port located at memory addresses 0 and 1 and the hardware control lines is meticulously detailed.

The description in the CBUS II manual is limited to the basic operation of the CBUS II cartridge as a cartridge emulator. It includes a program listing especially suited to the unique hardware of the cartridge. Both manuals include source code listings of the machine language portions for reference purposes. For advanced users, a full description of the CBUS II operation is included in the optional Technical Manual, along with a brief discussion on using the CBUS II as a cartridge program development tool.

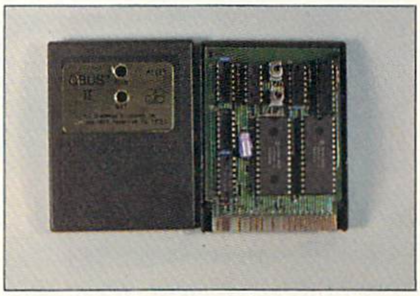
Several pamphlets were included with the package. These appear in a newsletter format. They deal with the specific nuances of particular cartridges for working with the CBUS system. One pamphlet lists the CBUS classifications for 75 commercially available C-64 cartridge programs.

The entire package is intended to let the average user copy and run cartridge-based software. The CBUS I and II manuals include detailed procedures on the proper use of their respective hardware and software.

As we went through the manuals, one area for improvement became evident. The largest manual was a bit over twenty pages. There was considerable amount of duplication of information across all three manuals. A significant percentage of each manual consists of overhead such as introductions, tables of contents, and copyright and warranty notices. We feel that both manufacturer and end user would benefit if the manuals were consolidated into one book.



CBUS-I allows the user to examine the contents of any Commodore 64 cartridge.



CBUS-II contains 16 kilobytes of static RAM with cartridge emulation circuitry.

THE SOFTWARE

If you opt for CBUS I Deluxe or the complete package, you will get all of the appropriate software on disk. This includes a menu generator program for use by either of the loader programs. Also included are two versions of Jim Butterfield's *Supermon* and a *Mini-Monitor* designed specifically for use with the CBUS package. The latter is a hybrid of machine language and BASIC. Overall we found the provided support programs to be effective but lacking in polish. But then again, their brevity is considered a virtue for this particular application.

WHAT IT'S GOOD FOR

The manufacturer places primary emphasis on the benefits of transferring cartridge software to disk. The saving of wear and tear on the cartridge port is one case in point. Additional benefit is to be obtained by the reduction of storage requirements, as several cartridges can be kept on one disk. Also touted was the convenience of being able to load programs directly from disk. We leave the evaluation of these considerations to the end user. We did feel that being able to rapidly swap cartridges held the edge in convenience.

We would have liked to see additional emphasis placed on the capabilities of the CBUS II as a cartridge development system. In this light, it would make an ideal companion to either of the ROM programmers in the accompanying reviews. Being able to try out a cartridge-based program in a total cartridge emulator without the need to continuously erase and reprogram a ROM is a convenience worth looking into.

Although the CBUS II cartridge includes sixteen kilobytes of RAM, do not expect to use it as expansion memory. The design of the CBUS II and its associated personality latch prevents simultaneous read and write operations to its RAM. The hardware can be configured to store data into the RAM or to read the data from it, but not both. To engage in two-way communication with CBUS II, your software would have to continuously reconfigure the personality latch.

CONCLUSION

The CBUS is an effective tool for manipulating cartridge firmware. The CBUS II is a valid cartridge development tool as a total cartridge emulator. If you already own an expan-

sion chassis for your C-64, then you may consider adding the CBUS I switches yourself. Although it does involve some soldering and cutting of printed circuit traces, it is not an overwhelming task. CBUS I sells for $34.95. CBUS II is $84.95. An additional $15.00 gets you CBUS I Deluxe, with the associated software on disk. The entire package can be had for $119.95. □

CARTRIDGE-MAKER-64: EPROM PROGRAMMING FOR THE MASSES

Custom Programming Group
47 Marchwood Road—Ste. 2A
Exton, PA 19341
Phone: 215-363-8840

The Cartridge-Maker-64 system simplifies EPROM programming so that anyone can do it. The entire chip selection process, along with the need to handle individual chips, has been eliminated. This operational simplicity does have its limitations. The system is designed to program only one type of EPROM, the 64 kilobit (eight kilobyte) 2764.

THE HARDWARE

A complete Cartridge-Maker-64 system consists of four components: the Cartridge-Maker-64 itself, a cartridge eraser, a blank Cartridge-64, and the user manual. These are available as a complete kit or as individual components.

The central component is the Cartridge-Maker-64, an EPROM programming module designed to program one or two type 2764 EPROMs. This 4 x 6 x 1¼" module fits into the computer's expansion port. An edge board connector at the back of the module accepts standard Commodore 64 program cartridges. A three-position toggle switch controls the power to this connector. The electrical status of the connector is indicated by a pair of light emitting diodes: a green one to indicate the presence of power and a red one to indicate programming activity.

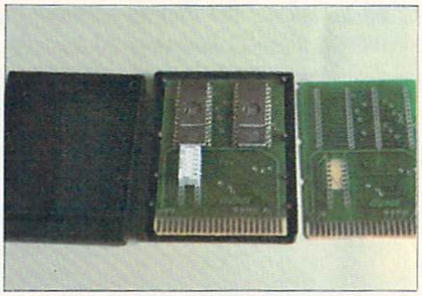
Hidden inside this black box is the PROM programming and control circuitry. This allows the single switch to electrically isolate the Cartridge-Maker-64 connector. When the switch is in the OFF position, it is safe to insert or remove a cartridge without turning off the computer. Also included in the package is an eight kilobyte operating system on ROM. This program takes care of all the cartridge reading, writing, and



Cartridge-Maker-64 lets you program up to two 2764 EPROMs at one time.



The built-in ultraviolet light source allows you to erase your EPROMs.



A Cartridge-Maker-64 cartridge. Center: EPROMs installed. Right: bare.

user interface tasks. The system is flexible enough to allow for the complete programming of a sixteen kilobyte cartridge without the need for a disk drive or cassette deck.

The cartridges designed for use with the Cartridge-Maker-64 are similar, but not identical, to commercial C-64 cartridges. Each Cartridge-64 comes fitted with a pair of type 2764 eight kilobyte EPROMs permanently soldered into place. The board's circuitry is designed to allow the 21 volt EPROM programming voltage to be applied by the Cartridge-Maker-64. The board is also fitted with a six position DIP switch for setting Cartridge-64 to any of four standard C-64 cartridge configurations. These consist of eight kilobyte cartridges at the $8000 or $E000 blocks or sixteen kilobytes at the $8000/$A000 or $8000/$E000 blocks. The EPROMs may be programmed individually or in pairs. This allows up to two Commodore 64 cartridges to be copied onto one Cartridge-64.

The Cartridge-Eraser is an ultraviolet light source designed to erase a Cartridge-64 in less than ten minutes. A safety interlock is built into the top of the case to prevent accidental exposure to the ultraviolet light. This light is intense enough to cause damage to the eyes. The switch is recessed inside the top cover well away from accidental activation. However, it is readily reached by prying fingers. Avoid this temptation and keep the eraser away from small children.

Although the hardware is specifically designed for PROM programming in the C-64 cartridge configuration, the system can be easily adapted for the programming of individual type 2764 PROM chips. A bare Cartridge-64 printed circuit board is available for $12. You will have to install your own six or eight position DIP switch for chip selection. Two 28 pin DIP sockets should also be soldered onto the board. These should be of the zero insertion force (ZIF) construction. When complete you will be able to easily program 2764s, individually or in pairs, with the Cartridge-Maker-64 system.

THE FIRMWARE

Cartridge-Maker-64 comes with its own built-in menu-driven software in an autostart ROM. The opening menu lets you select the screen colors. This is followed immediately by the main menu with the nine system selections. The first of these is the *Cartridge to Cartridge* utility. This function will automatically identify the original cartridge type and read

the data into memory. You will then be instructed in the proper switch settings for the Cartridge-64. Simply follow the prompts for switching cartridges and setting the program control switches.

Cartridge-64 will not work with cartridges designed for switching between external ROM and C-64 internal ROM or RAM. These cartridges are equipped with onboard hardware latches that Cartridge-64 does not emulate.

The next two main menu choices deal with the placing of BASIC programs into cartridge. This will let you create an autostart cartridge without any knowledge of machine language. When the computer is turned on the program will be automatically downloaded into the normal BASIC program area and RUN from there. The program can even be STOPed, LISTed, and modified as desired. BASIC programs in cartridge are limited to a maximum length of 8,000 bytes by the Cartridge-Maker-64 system. The presence of the Cartridge-64, which contains the BASIC program, also reduces BASIC program space to thirty kilobytes. Placing a BASIC program into a cartridge is a simple way to create an autostart program which will not require a disk drive or cassette deck to LOAD and RUN.

The next two menu choices allow for the transfer of data between cartridge and memory. These are the basic 2764 programming modes. Locations $6000-$9FFF serve as a buffer area in the C-64. You will have to LOAD these buffer areas with the appropriate data before using these menu choices.

The next four main menu selections deal with the transfer of data from disk or tape to or from a C-64 cartridge. Cartridge-Maker-64 disk files are sequential files. Tape files are program files. Included in all files is information on the original cartridge type in addition to the program data.

The last menu choice lets you exit to BASIC. A simple SYS call will enable you to reactivate the Cartridge-Maker program.

THE MANUAL

All of the Cartridge-Maker-64 system features are thoroughly described in the thirty-page user's manual. The presentation is geared to the user with a minimum of technical background. However, some understanding of the C-64 memory structure is needed to obtain the maximum benefit from this system. Complete details of the Cartridge-Maker file structures, as well as descriptions of the various C-64 memory maps, are included.

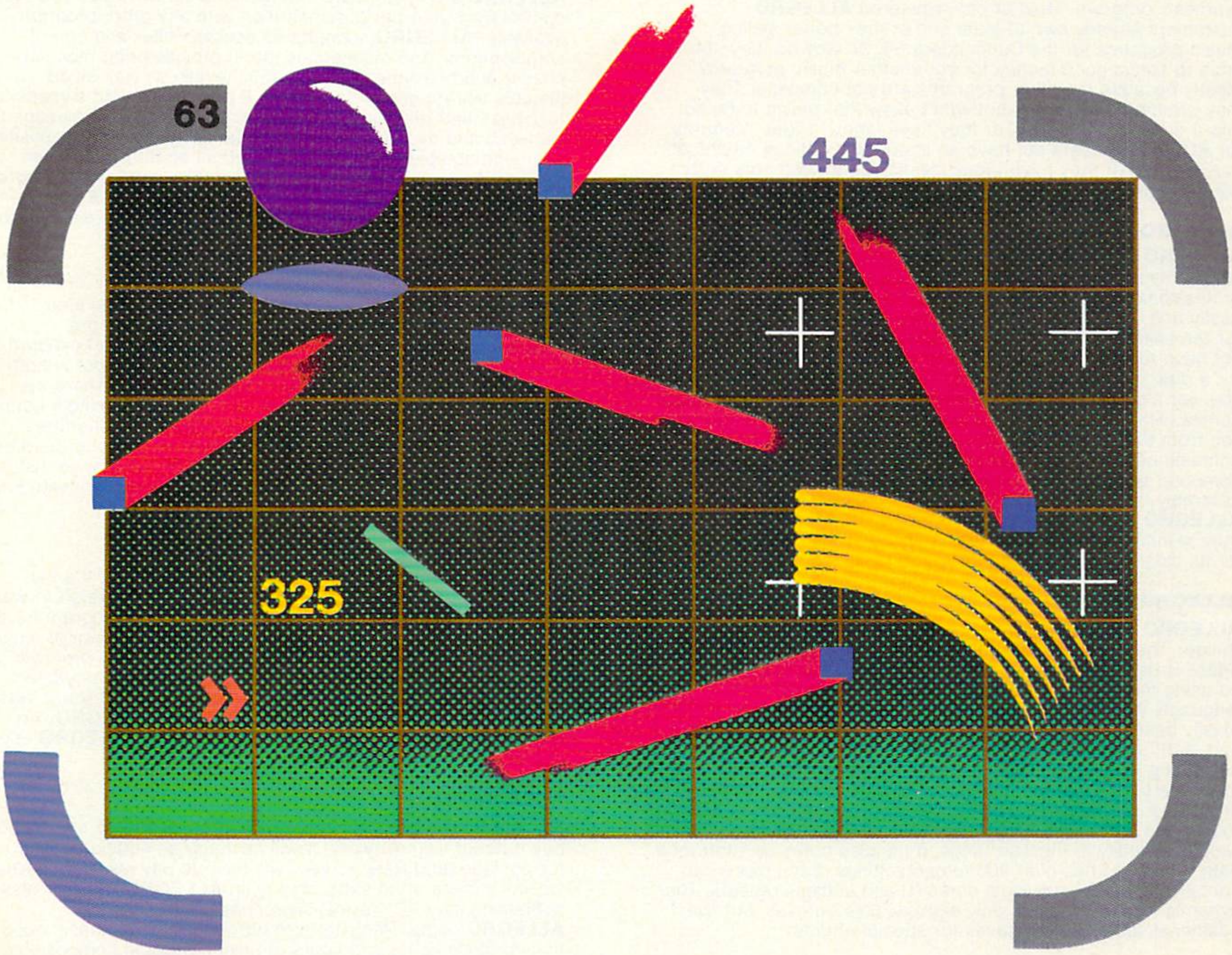
CONCLUSION

The Cartridge-Maker-64 system is certainly an easy-to-use EPROM programmer. This ease of use incurs a substantial penalty with regard to overall flexibility. The system is limited at best to the programming of a single type of EPROM. Although considerable emphasis is placed on the copying of cartridges, it is unlikely that this will be cost effective. An entire system, including the eraser and programming modules as well as a single sixteen kilobyte module, sells for $189. Additional sixteen kilobyte modules, with EPROMs, are $25 or five for $115. The system can be used to minimize program development times by allowing BASIC programs to be placed into cartridges. This storage medium is suitable for custom applications in unattended locations where the harsh environment may preclude the use of a disk drive.

AUTHOR'S NOTE

A machine language monitor (MLM) is required to obtain the maximum benefit from these products. Several excellent MLMs are available in the public domain. If you have not yet come across these programs, I will provide a copy of my MLM disk for a $12 copying and distribution fee. This self-documenting disk includes *Extramon*, *Micromon*, and some disk editing utilities. Send your request to Morton Kevelson, P.O. Box 260, Homecrest Station, Brooklyn, NY 11229. ☐

SPEEDING PIXELS



Some Assembly Language Help for Bit-Map Graphics

By Dale W. Rupert

We saw last month that BASIC is very sluggish when it comes to manipulating the 64,000 pixels in high resolution bit mapped graphics. This month we will develop two assembly language routines to help speed things up. The first routine does nothing more than clear the 8000-byte bit map memory. The second routine performs the calculations necessary to locate a pixel on the screen and puts it there.

When we are finished this month, we will have a somewhat cumbersome interface to BASIC. We must still POKE X and Y values in order to plot the point (X,Y). We will not be able to draw lines instantly between any two points. But we will have the groundwork on which to build other capabilities in the future. We will create some very diverse assembly language modules. Consequently, you may find this article useful as a machine language tutorial.

PRELIMINARIES

The BASIC program on page 89 (Listing 1) is similar

"You've Tried The Rest... Now Try The Best!"

A clever advertising slogan? Perhaps. But we at Artworx have found that phrase to apply to our **ALLEGRO** advanced music synthesis program. Most of our registered **ALLEGRO** customers already own at least one of the "better selling" music programs for the Commodore 64. So why do they continue to spend good money for yet another music program? Mostly because the other programs are not complete. They may produce nice music, but won't allow that music to be put into a computer program; or they have fancy "slider" controls for ADSR envelopers but have an impossibly difficult means of music entry; or they have fancy joystick controlled note entry but don't play worth a darn. Not so with **ALLEGRO.**

ALLEGRO IS DIFFERENT

ALLEGRO is a complete music synthesis program. By that we simply mean that **ALLEGRO** does everything required to obtain professional quality electronic music. It does it well, it does it easily and it does it intelligently - with no distracting gimmicks for the sake of gimmicks. And it does it all on one disk. You will not have to spend an extra $35 for a "synthesizer" disk or $25 for a disk that allows you to play music from the keyboard. And you won't be frustrated by incomplete instructions - **ALLEGRO** comes with over 70 pages of *extensive* documentation taking you from theory to application and all points in between. It even contains helpful information on hooking up your stereo to the computer and including your music in your own BASIC or assembly language programs. One review stated that **ALLEGRO** "...is a technical work of art. The demo songs are truly stunning, the 74-page manual is worth the purchase price for its clear discussions of music and digital sound."

ALLEGRO'S FORTE EDITOR

ALLEGRO consists of four parts: the Editor, the Instrument Builder, the Registers Control and the Modulations Control. The Editor is used to enter and edit your music. This is accomplished using the specially developed FORTE music notation language. It is the fastest method of inputting a musical score -easily beating out the the best of the joystick controlled "note and staff" methods, and very simple to use for modifying or correcting your music. With the FORTE editor, you will be able to create music that can use up to 85 different instruments in one piece, or choose any of the 50 pre-programmed instruments that come with the program. **ALLEGRO** gives you the widest flexibility in musical range, it handles notes as short as a 64th triplet and has over 900 tempo settings. Each piece can contain over 5800 individual notes (!) and endless repeats. The dynamic range spans not only eight audible octaves, but five additional *sub audible* octaves for special effects!

INSTRUMENTS

Every music program comes with pre-programmed instruments, but **ALLEGRO** has over 50 ranging from harpsichord to drums to bagpipes. Every music program allows you to change instruments within a piece, but **ALLEGRO** gives the user the flexibility of selecting any of up to 85 instruments. And any of these instruments can be modified or replaced by another group of 85 user-defined instruments. Coupled with **ALLEGRO's** use of registers and modulations (using over 20 oscillators available within the Commodore 64), all sorts of unique instrumental sounds are at your disposal.

SPECIAL EFFECTS AND FEATURES

ALLEGRO's Registers and Modulations controls give you more control over your performance than with any other program available. **ALLEGRO** supports all possible filter and cut-off combinations, and resonances too. It provides ring modulator, phaser and heavymetal effects (250 levels) as well as adjustable vibrato and siren effects. It has pulse width sweeps for fuzz bass and realistic piano sounds as well as filter sweeps for wah-wah and dynamic timbre changes. And once a composition has been completed, it can be stored as an interrupt driven PASSKEY file which can be called from the user's own program by a one line instruction. A PASSKEY file can hold a maximum of 41 songs and requires only 4K from BASIC regardless of the amount of music that it contains!

PERFORMANCE FEATURES

ALLEGRO can turn your Commodore 64 into a real time musical instrument allowing you to play along with pre-programmed background rhythms directly from the keyboard. In fact, you will find that the **ALLEGRO** keyboard layout is compatible with certain add-on piano style keyboards. There are two performance modes: monophonic, for playing single notes along with a background track, and polyphonic, using three voices and simultaneous keystrokes allowing real chords to be played. During performance, you will be able to change instruments, octaves, vibrato, staccato, wah wah and more with just one keystroke!

SUMMING IT UP

As you can tell, we're pretty keen on this program. It's the perfect program for the beginner because of its straightforward layout and clear documentation. It's the perfect program for the knowledgeable user because of its flexibility and ease of music entry and transcription, and it's the perfect (if only) program for the professional because of its completeness and capabilities—it has already been used to create the music for at least two best-selling Commodore programs. With **ALLEGRO,** Artworx has provided all the necessary tools, the **ALLEGRO** user is limited only by imagination.

FREE OFFER!

No, **ALLEGRO** does not come free. It costs $39.95 and if you buy it direct from Artworx, it will cost you an additional $2.00 for postage (and New Yorkers will have to pay sales tax). What we will include at no extra cost is an **ALLEGRO** sampler disk containing over 40 musical selections contributed by **ALLEGRO** users. Ranging from baroque to punk, these selections provide hours and hours of unique musical composition. No, you will not find "Home On The Range" or "Jingle Bells" on this disk! There is one catch, however: we must receive your order before July 31, 1985. Call toll-free: 1-(800)-828-6573 or send check or money order. VISA or MasterCard gladly accepted.

Artworx®

ARTWORX Software Co., Inc., 150 North Main St., Fairport, NY 14450

Please Send Me *ALLEGRO*

Enclosed is my check for $_______. Bill my ☐VISA ☐MasterCard

NUMBER________________EXP. DATE________

Signature ____________________

Name (please print) ____________________

Address ____________________

City____________State________Zip_______

ARTWORX Software Co., Inc., 150 North Main St., Fairport, NY 14450

Reader Service No. 170

to the program we discussed last month. To begin, let's review the concepts and procedures involved in high resolution bit map graphics.

The bit map consists of 8000 bytes beginning in RAM at location 8192. Each of those 64,000 bits corresponds to a specific dot or pixel (picture element) on the screen. If a bit is set to one, the pixel is one color, and if a bit is reset to zero, its pixel is another color. The screen appears as an array of 320 (horizontal) by 200 (vertical) pixels.

The 1000 bytes of normal screen memory from locations 1024 to 2023 each store two nybbles of color data. The color resolution is 64 bytes. That means each byte of color data applies to 64 pixels on the screen. The data in location 1024 defines the possible colors for the 64 pixels in the upper left (eight-by-eight) corner of the screen. Similarly, the data in location 2023 defines the colors for the lower right corner. There are 16 possible colors to choose from, but each eight-by-eight block of pixels will consist of any two of those colors.

Some examples should clarify all this. Assume location 1024 contains the value $24 (i.e. decimal 36—the "$" signifies a hexadecimal value). The possible colors of the 64 pixels in the upper left corner of the screen are red and purple. To understand this, refer to the color values listed in Appendix D of the *Programmer's Reference Guide (PRG)* or Appendix G of the *User's Guide.* Red has a value of 2; purple has a value of 4. The hexadecimal number with one nybble of 2 and the other nybble of 4 is the value to select the colors red and purple.

To select light red and light blue for the lower right corner, put 174 ($AE) or 234 ($EA) into location 2023. Light red has a value of 10 ($A), light blue has a value of 14 ($E). The hexadecimal color code is then $AE or $EA. To convert to the decimal equivalent, multiply the first value by 16 and add it to the second: 16*10 + 14 equals 174.

The first color code (most significant nybble) corresponds to the pixels whose bit map memory bits are set to one, and the second color code is for the pixels whose bits are reset to zero.

The chart on page 125 of the *PRG* shows the rather complicated addressing structure of the bit map memory. The formulas for locating the correct memory location and the bit within that location for a particular pixel are contained in lines 400 to 420 of the BASIC program in Listing 1.

We saw last month that to use bit map graphics mode we must define the starting location of the bit map memory by setting bit 3 of VIC-II register 24. Then we must set bit 5 of VIC-II register 17 to enter bit map mode. Those functions are performed in lines 40 and 50 of the BASIC program in Listing 1. The user-defined functions FNSB and FNRB are "set bit" and "reset bit" functions. The statements MM=VV+24 : POKE MM, FNSB(3) set bit 3 of location MM, where MM has the address of VIC-II register 24.

The program in Listing 1 may be run without creating the machine language routines we are discussing here. Once you have assembled the two programs with filenames CLSCRN.O and PLOT.$C.O and have them available on disk, you may delete lines 65 and 265 of Listing 1. Then the two machine language programs will be used instead of their BASIC equivalents. (If you prefer to get CLRSCR.O and PLOT.$C.O up and running without assembling them, we've provided *Flankspeed* listings of each on page 89.)

It takes an interminably long time (roughly thirty seconds!) for BASIC to clear the 8000 bytes of bit map memory using a FOR-NEXT loop. In this type of application, machine language really shines. Let's see what it takes to speed up the screen-clearing process—by a factor of 135 or more!

NOW YOU SEE IT

The CLRSCR.O assembler listing puts zeroes into 8000 consecutive locations beginning at address 8192 ($2000). The programs in this article were assembled with the *Merlin 64* macro assembler. The main differences between listings from different assemblers are the assembler directives, more commonly known as pseudo-ops. Pseudo-ops look somewhat like the genuine-ops, the mnemonic op-codes of the 6510. Actually they're instructions to the assembly itself, not to the processor.

Lines 6 and 10 through 14 of the CLRSCR.O listing contain pseudo-ops. ORG is fairly standard. It tells the assembler the ORiGin of the program, that is, where this program will reside in memory. The EQU assigns a value to a label. In this case, it tells the assembler that a reference to PAGE0 actually refers to memory location $FB. Neither ORG nor EQU cause any object code to be generated. Again, they are instructions to the assembler.

Assembled Version of CLRSCR.O

```
                1    *
                2    * RUPERT REPORT #19
                3    * HI-RES GRAPHICS
                4    * CLEAR SCREEN ROUTINE
                5    *
                6              ORG    $33C         ;TBUFFR (828)
                7    *
033C: 4C 44 03  8              JMP    START
                9    *
                10   PAGE0     EQU    $FB          ;USE 2 BYTES
033F: 00 20     11   BASE      DA     $2000        ;SCRNMEM (8192)
0341: 1F        12   BLOKS     DFB    31           ;8000=31*256
0342: 40        13   XTRA      DFB    64           ;     +64
0343: 00        14   FILL      DFB    0            ;SCRN FILL CHAR
                15                                 ;
                16   START
0344: AD 3F 03  17             LDA    BASE          ;BASE LSB
0347: 85 FB     18             STA    PAGE0         ;PAGE ZERO
0349: AD 40 03  19             LDA    BASE+1        ;BASE MSB
034C: 85 FC     20             STA    PAGE0+1
034E: AE 41 03  21             LDX    BLOKS         ;NO. BLOCKS
0351: A0 00     22             LDY    #0            ;256 BYTES
0353: AD 43 03  23             LDA    FILL
0356: 88        24   LOOP      DEY
0357: 91 FB     25             STA    (PAGE0),Y
0359: D0 FB     26             BNE    LOOP            ;NO
035B: E6 FC     27             INC    PAGE0+1        ;DO NEXT 256
035D: CA        28             DEX                 ;ANY MORE?
035E: 30 07     29             BMI    DONE            ;NOPE
0360: D0 F4     30             BNE    LOOP          ;DO MORE BLOCKS
0362: AC 42 03  31             LDY    XTRA          ;DO EXTRAS
0365: D0 EF     32             BNE    LOOP          ;FILL EXTRAS
                33   DONE
0367: 60        34             RTS                 ;BACK TO BASIC
                35   END
```

The DA and DFB pseudo-ops, on the other hand, do generate code. The assembler takes the values following each of these and places them in consecutive memory locations. DA stands for Define Address, and the value following it is stored in the standard "least significant byte (LSB) first, most significant byte (MSB) last" sequence. DFB allows the programmer to define one or more bytes to be put into memory. Both of these pseudo-ops also allow the programmer to assign labels to these values, similar to variables and the LET statement in BASIC.

A block of two hundred fifty-six bytes is referred to as a page. The eight thousand bytes of bit map memory corresponds to thirty-one pages plus 64 additional bytes. This routine uses the Y register to cycle from 255 down to 0, pointing to individual addresses on a page, while the X register keeps track of how many pages have been cleared. Once the 31 pages have been cleared, the number of extra bytes (64) is loaded into the Y register, and those bytes are cleared. Let's look at the details.

The DA statement in line 11 puts $00 into address BASE and $20 into the next location, which is BASE+1. For example, when this program is assembled, the JMP statement in line 8 consisting of three bytes is put into locations $33C, $33D, and $33E (since the ORG was $33C). Consequently BASE corresponds to address $33F and BASE+1 refers to address $340.

Line 17 (LDA BASE) loads the accumulator with the $00 it finds at address BASE. Line 18 puts this value into page zero location $FB (PAGE0). Similarly, the value at address BASE+1 is put into location $FC by lines 19 and 20. The X, Y, and A registers are loaded with their initial values of 31, 0, and 0 in lines 21 through 23. The first statement of the main loop at line 24 decrements the Y register, so Y now contains $FF.

The main loop of this program uses indirect indexed addressing. This mode of addressing requires that the address to be indexed is stored on page zero. That is the purpose of lines 17 through 20. We have chosen page zero addresses $FB and $FC to store the page address of each block as it is cleared. According to page 316 of the *PRG*, addresses $FB through $FE are available to us.

Indirect indexed addressing is among the more complicated addressing modes. Let's examine the instruction in line 25 of the CLRSCR.O listing:

```
STA (PAGE0),Y
```

This instruction says to store the contents of the accumulator at the address which is indicated by the contents of the page zero locations PAGE0 and PAGE0+1, as indexed by the Y register. Phew!! Let's try this step by step.

The effective address at which the data in the accumulator is stored is determined through a roundabout sequence of events. First recall that PAGE0 is a label associated with the page zero address $FB, as defined in line 10. The processor reads the two bytes stored at locations $FB and $FC. Initially these will be $00 and $20 respectively, as we saw above. These two bytes are interpreted to be the address $2000. Next, the processor takes the value in the Y register (initially $FF) and adds it to the $2000. The effective address is consequently $20FF. The 0 which is in the accumulator is stored at address $20FF, thereby clearing that memory location.

According to the table on page 417 of the *PRG*, the entire sequence just described takes a grand total of 6 microseconds. You can see this by reading across the STA row to the "(Ind.) Y" column. There you see the hexadecimal value ($91) for this mode of the STA instruction. You also see that the instruction consists of 2 bytes (#) and requires 6 machine cycles (N). Each machine cycle on the Commodore 64 is roughly 1 microsecond (one-millionth of a second). All the address fetching, calculating, and storing described in the previous paragraph takes less than 1/160,000th of a second!

Now that location $20FF has been cleared, the program continues at line 26. The statement BNE LOOP means "check the zero flag, and if it's not set, jump to the statement labeled LOOP." Let's discuss this concept of flag setting and checking.

FLAG WAVING

The zero flag is one bit inside the processor which is updated whenever certain operations are performed by the processor. The STA instruction does not cause

the zero flag to be updated. This is indicated by the "–" under the Z in the Condition Codes column on page 417 of the *PRG*. This information is repeated in larger letters on page 251 of the *PRG*.

If you look up the DEY instruction on page 242 of the *PRG*, you will see that the N and the Z flags are checked. This means that after the processor performs a DEY instruction, it looks at the result of the operation and sets those two flags accordingly. The N flag is set whenever the most significant bit (MSB) of the result is set. ("Set" means "equal to one.") The Z flag is set whenever the result of an operation is zero.

The value of Y was initially 0, and the first DEY instruction decremented it to $FF. Since $FF is not equal to zero, the Z flag will not be set. (That is to say, it will be reset or cleared.) Consequently, the BNE ("Branch if Not Equal to zero") instruction sends the processor back to address LOOP. A quasi-BASIC translation of the DEY : BNE LOOP instruction pair might be

```
Y = Y-1 : IF Y<>0 THEN GOTO LOOP
```

Read the description of the BNE instruction on page 238 of the *PRG*. Keep in mind that the state of a flag is determined by the last instruction which affected that flag.

The three instructions in the main loop beginning in line 24 of the CLRSCR.O listing will be executed 255 times, with Y being decremented from 255 to 1 and addresses $20FF through $2001 being cleared. On the 256th time that line 24 is executed, Y is decremented to zero, thereby setting the Z flag. Address $2000 will be cleared, and the conditions for the BNE instruction to branch back to LOOP are not met. Execution continues with line 27.

The page address stored in location PAGE0+1 is incremented from $20 to $21. Now addresses $21FF through $2100 will be cleared. The X register is decremented in line 28. It is keeping count of the number of pages which still must be cleared. Since X is decremented from 31, the result is 30 and therefore the N flag is not set. The BMI ("Branch on a MInus result") statement looks at the N flag. Results which have an MSB of one are considered "minus." Thus $80 through $FF (128 through 255) are negatives. $00 through $7F (0 through 127) are positives or "plus" results.

Since the N flag is not set, execution continues at line 30. The Z flag is not set (since X was not decremented to 0), and line 30 branches back up to the main loop to clear the next page of memory. A translation of the instructions DEX : BMI DONE : BNE LOOP might be

```
X = X-1 : IF X>127 THEN GOTO DONE
IF X<>0 THEN GOTO LOOP
```

Keep in mind that X is an eight-bit quantity and decrementing it from 0 gives a result of 255.

On the 31st time the DEX in line 28 is executed, the Z flag will be set, since X becomes zero. At this point, the 31 ($1F) whole pages from $2000 to $3EFF have been cleared. The BMI instruction in line 29 is still bypassed, since X is not negative yet. The BNE in line 30 is also bypassed, since X now equals zero. The Y register is loaded with the number of additional bytes ($40) to be cleared. The page address has already been incremented to $3F00 in line 27.

Now as the main loop is repeated, addresses $3F3F through $3F00 are filled with zeroes. Line 28 decrements X to $FF, and the BMI instruction sends the program to its conclusion. The RTS instruction returns control to BASIC.

HOW FAST?

The instructions in the main loop must be executed 8000 times. Is the increase in speed really worth all that we've just been through? The results of some simple timing tests are very convincing. Clearing 8000 bytes of RAM in BASIC using a FOR-NEXT loop require 1766 jiffies (sixtieths of a second). This machine language routine clocks in at a grand total of 13 jiffies, and that includes the time required for BASIC to read the object code file into memory from the disk. The routine itself takes only 6 jiffies once it is in memory.

Hard to believe, but true. The machine language program is over 135 times faster than BASIC and nearly 300 times faster if it is already in memory. That's what makes assembly language worth it all–aside from being an exciting intellectual challenge!

CREATING THE PLOT

We will not go through the second assembly language program in as much detail as the first. The purpose of the second assembly language program is to process the specified values of X and Y and to light the appropriate pixel on the screen. Refer to the PLOT.$C.S listing for the following discussion.

The BASIC program in Listing 1 POKEs values into memory locations XVAL, XVAL+1, and YVAL for X and Y. X is a two byte quantity between 0 and 319. Y is a one byte quantity from 0 to 199. The first task of PLOT.$C.S is to make sure that the X and Y values are in the proper range. If either is out of range, it is changed to the maximum value.

Beginning at line 43, the program evaluates each term of the expression

```
Z = 320*INT(Y/8) + (Y AND 7) + 8*INT(X
/8) + 8192
```

which we saw last month gives the address of the memory cell corresponding to the point (X,Y). Taken a piece at a time, the task is not as formidable as it might seem.

The most difficult problem is to multiply INT(Y/8) by 320. The procedure is to first calculate 8*INT(Y/8) and to multiply that result by 40. Finding 8*INT(Y/8) is simply a matter of dropping off (truncating) the three least significant bits of Y. This is done with the statement in line 52, AND #$FB. The accumulator contains

the one byte quantity Y. The "#" implies the immediate addressing mode. The processor takes the quantity $F8 and performs the logical AND function between it and the accumulator's value, leaving the result of the calculation in the accumulator.

We are using a masking operation here. The three least significant bits of $F8 are zeroes and the other five bits are ones. ANDing any quantity with zero gives zero while ANDing any quantity with one leaves the quantity unchanged. Consequently, the five most significant bits of Y are unchanged and the three least significant bits become zero by the operation (Y AND $F8). This is equivalent to the value 8*INT(Y/8).

The result is then multiplied by forty. Multiplication by powers of two is relatively easy in machine language. It amounts to little more than shifting a binary quantity one place to the left. Unfortunately, forty is not an even power of two. It is equal to the sum of two separate powers of two, namely eight and thirty-two. We will use the fact that 40*Z equals (8*Z + 32*Z), where Z presently equals 8*INT(Y/8). The result will be a two byte quantity. The MSB will be in location Z+1 and the LSB will be in location Z.

The ROL (rotate left) instruction in line 61 is used to

Assembled Version of PLOT.$C.O

```
                1   *
                2   *   RUPERT REPORT #19
                3   *   ASSEMBLER LISTING 3
                4   *   HIGH RESOLUTION PLOTTER
                5   *
                6           ORG   $C000      ;START AT 49152
                7   *
C000: 4C 08 C0  8           JMP   START
                9   *
C003: 3F 01     10  XVAL    DA    319        ;2 BYTE:LSB,MSB
C005: 00        11  YVAL    DFB   0          ;1 BYTE
C006: 00 20     12  BASE    DA    $2000      ;BIT-MAP MEMORY
                13  Z       EQU   $FB        ;4 BYTES ON PAGE 0
                14  *
                15  START
                16  ********************************
                17  *     MAKE SURE X < 320        *
                18  ********************************
C008: AD 04 C0  19          LDA   XVAL+1     ;CHECK XVAL MSB
C00B: F0 13     20          BEQ   OKX        ;OK IF 0 (X<256)
C00D: A9 01     21          LDA   #1         ;--- X > 255 ---
C00F: 8D 04 C0  22          STA   XVAL+1     ;MAKE MSB=1
C012: AD 03 C0  23          LDA   XVAL       ;LSB MUST BE<64
C015: 30 04     24          BMI   FIXX       ;FIX IF NOT
C017: C9 40     25          CMP   #$40       ;LSB<64?
C019: 30 05     26          BMI   OKX        ;YES
C01B: A9 3F     27  FIXX    LDA   #$3F       ;USE MAX LSB
C01D: 8D 03 C0  28          STA   XVAL
                29  OKX                      ;NOW XVAL<320
                30  ********************************
                31  *     MAKE SURE Y < 200        *
                32  ********************************
C020: AD 05 C0  33          LDA   YVAL
C023: 10 0B     34          BPL   OKY        ;Y < 128
C025: 29 7F     35          AND   #$7F       ;Y=Y-128
C027: C9 48     36          CMP   #$48       ;(200=128+$48)
C029: 30 05     37          BMI   OKY        ;Y<200
C02B: A5 C7     38          LDA   199        ;USE Y MAX
C02D: 8D 05 C0  39          STA   YVAL
                40  OKY                      ;NOW YVAL<200
                41
                42
                43  *================================*
                44  *  Z=40*(8*INT(Y/8)) + (Y AND 7) *
                45  *     + 8*INT(X/8) + BASE        *
                46  *================================*
                47
                48  ********************************
                49  *  GET Z = 8*INT(Y/8)          *
                50  ********************************
C030: AD 05 C0  51          LDA   YVAL       ;TRUNCATE
C033: 29 F8     52          AND   #$F8       ;   3 LSbits
C035: 85 FB     53          STA   Z          ;WORK AREA
C037: A9 00     54          LDA   #0
C039: 85 FC     55          STA   Z+1        ;MSB=0
                56  ********************************
                57  *  GET Z = 40*Z (2 BYTE QTY.)  *
                58  ********************************
C03B: A5 FB     59          LDA   Z
C03D: 18        60          CLC
C03E: 2A        61          ROL   A
C03F: 26 FC     62          ROL   Z+1        ;2*Z
C041: 2A        63          ROL   A
C042: 26 FC     64          ROL   Z+1        ;4*Z
C044: 2A        65          ROL   A
C045: 26 FC     66          ROL   Z+1        ;8*Z
C047: 85 FD     67          STA   Z+2        ;--LSB 8*Z--
C049: A5 FC     68          LDA   Z+1
C04B: 85 FE     69          STA   Z+3        ;--MSB 8*Z--
C04D: A5 FD     70          LDA   Z+2        ;LSB 8*Z
C04F: 2A        71          ROL   A
C050: 26 FC     72          ROL   Z+1        ;16*Z
C052: 2A        73          ROL   A
C053: 26 FC     74          ROL   Z+1        ;32*Z
C055: 18        75          CLC
C056: 65 FD     76          ADC   Z+2        ;LSB 32*Z+8*Z
C058: 85 FD     77          STA   Z+2
C05A: A5 FC     78          LDA   Z+1
C05C: 65 FE     79          ADC   Z+3        ;MSB 32*Z+8*Z
C05E: 85 FE     80          STA   Z+3
                81  * RESULTS IN Z+2(LSB) AND Z+3(MSB)
                82  ********************************
                83  *  GET Z = (Y AND 7) + Z       *
                84  ********************************
C060: AD 05 C0  85          LDA   YVAL
C063: 29 07     86          AND   #7         ;Y AND 7
C065: 18        87          CLC
C066: 65 FD     88          ADC   Z+2        ;ADD TO Z
C068: 85 FD     89          STA   Z+2        ;LSB
C06A: A5 FE     90          LDA   Z+3
C06C: 69 00     91          ADC   #0         ;ADD CARRY
C06E: 85 FE     92          STA   Z+3        ;MSB
                93  ********************************
                94  *  GET Z = 8*INT(X/8) + Z      *
                95  ********************************
C070: AD 03 C0  96          LDA   XVAL       ;LSB
C073: 29 F8     97          AND   #$F8       ;8*INT(X/8)
C075: 18        98          CLC
C076: 65 FD     99          ADC   Z+2        ;LSB Z
C078: 85 FD     100         STA   Z+2
C07A: AD 04 C0  101         LDA   XVAL+1     ;X MSB-NO CHANGE
C07D: 65 FE     102         ADC   Z+3
C07F: 85 FE     103         STA   Z+3        ;MSB Z
                104 ********************************
                105 *  GET Z = BASE + Z            *
                106 ********************************
C081: AD 06 C0  107         LDA   BASE       ;LSB
C084: 18        108         CLC
C085: 65 FD     109         ADC   Z+2        ;LSB Z
C087: 85 FD     110         STA   Z+2
C089: AD 07 C0  111         LDA   BASE+1     ;MSB
C08C: 65 FE     112         ADC   Z+3        ;MSB Z
C08E: 85 FE     113         STA   Z+3
                114 ********************************
                115 * FIND BIT = 7 - (X AND 7)     *
                116 ********************************
C090: AD 03 C0  117         LDA   XVAL       ;X LSB
C093: 29 07     118         AND   #7         ;(X AND 7)
C095: 85 FB     119         STA   Z          ;SAVE IT
C097: A9 07     120         LDA   #7
C099: 38        121         SEC              ;NORMAL SUBTRACT
C09A: E5 FB     122         SBC   Z          ;BIT=7-(X AND 7)
                123 ********************************
                124 *    SET BIT IN LOCATION Z     *
                125 ********************************
C09C: AA        126         TAX              ;BIT# IN X
C09D: 38        127         SEC              ;ROTATE CARRY...
C09E: A9 00     128         LDA   #0         ; X TIMES INTO A
C0A0: 2A        129 LOOP    ROL   A
C0A1: CA        130         DEX              ;DONE YET?
C0A2: 10 FC     131         BPL   LOOP       ;NO
C0A4: 85 FC     132         STA   Z+1        ;SAVE IT
C0A6: A0 00     133         LDY   #0         ;NO OFFSET
C0A8: B1 FD     134         LDA   (Z+2),Y    ;GET SCREEN BYTE
C0AA: 05 FC     135         ORA   Z+1        ;SET PROPER BIT
C0AC: 91 FD     136         STA   (Z+2),Y    ;UPDATE SCREEN
C0AE: 60        137 END     RTS
```

rotate the LSB of Z which is in the accumulator. The most significant bit which falls off the left side of the accumulator goes into the carry flag inside the processor. The next ROL is performed directly on memory address Z+1. The carry bit is rotated into the least significant bit position of Z+1, and all the zeroes in Z+1 get shifted one place to the left. Notice that the carry bit was cleared in line 60 before the process was begun.

After three repetitions, Z and Z+1 store a two byte quantity which is eight times its original value. We will use this value later so it is copied into locations Z+2 and Z+3 for safe keeping. Two more repetitions gives thirty-two times the original value of Z and Z+1. Lines 75 and 80 add this quantity to the intermediate values at Z+2 and Z+3 and put the results into Z+2 and Z+3, thus giving forty times the original value of 8*INT(Y/8).

At line 113 of the PLOT.$C.S listing, the address corresponding to the pixel at location (X,Y) is stored in Z+2 and Z+3 (LSB, MSB). Lines 115-122 calculate the proper bit value (0 through 7) corresponding to that pixel.

The final segment of the program at line 124 shifts a single one into the proper bit position of the accumulator. This result is ORed with the value in the selected bit map memory location, thereby turning on the pixel at position (X,Y).

Calling this machine language subroutine to calculate and plot each point is still roughly twice as fast as using the BASIC subroutine in lines 400 through 430 of Listing 1 to do the same thing. The program in Listing 1 checks the keyboard buffer index after each point is plotted. The INKEY$ routine does not work in this bit map graphics mode, so a more fundamental approach was required to interrupt the program. Remember that if the program stops in bit map graphics mode, you can blindly enter GOTO 300 to return the screen to text mode. Once the patterns have been drawn, the program waits for any key to be pressed before returning to normal mode.

The best ways to learn assembly language are to read books and magazine articles about it, study examples of programs written by others, and (most important) get an editor/assembler and use it. Two books that I have found useful are *6502 Software Gourmet Guide & Cookbook* by Robert Findley (Scelbi) and *Programming the 6502* by Rodnay Zaks (Sybex).

We will wrap things up with three challenges you might enjoy. First, modify the CLRSCR.O listing so that it fills the color memory (1024 through 2023) with a specified color code in addition to clearing the bit map memory. Second, write a machine language routine to draw a line between endpoints specified by the user. (Perhaps start with horizontal and vertical lines only.)

Third, see how quickly you can analyze this bug that took me a couple of hours to eliminate. Swap the DEY and STA instructions in lines 24 and 25 of CLRSCR.O. LOOP is now the label for the STA instruction. The program still works almost perfectly. In fact, it works fine for 7999 of the 8000 bytes. Can you figure out where it fails?

As a follow-up to the March *Rupert Report* on the DOS Wedge, James Borden (Carlisle, PA) sent the following additions and a correction. To change the disk drive number, type @#d <RETURN> where d is the new drive number (8 or 9). You can pause a directory listing by tapping the space bar. Pressing it again continues the scrolling or pressing <RUN/STOP> stops it. Mr. Borden suggested that after pressing <RUN/STOP>, you may move the cursor up the left column to the line containing the desired filename and type "/", "%" or <up arrow> then <RETURN>. That way you may load and run a program without having to type its name. (You don't have to erase the numbers in the file size column either.) Mr. Borden correctly pointed out that entering LOAD"W",8:RUN will *not* automatically load and run the program "W". Instead of typing RUN after the colon, you must hold the shift key and press the <RUN/STOP> key. Thanks for the feedback.

Until next month, keep those programs speeding along. Inject a bit of machine language into them. □

SEE PROGRAM LISTINGS ON PAGE 89

BRAINFRAME

An Exercise in Logic for the C-64

By Norm Vogel

Submitted for your approval is a new version of the venerable logic puzzle, *Mastermind* (*"MM"*)™. I wrote the original version of this program when *MM* first came out, but since the advent of the C-64, it was modified extensively to take advantage of the 64's color and sound capabilities. The result is *Brainframe*–a program which I feel is superior to other versions of this game currently available, and not just another number guessing game.

After the title screen clears, the instructions are printed. The program then prompts for the number of digits to solve (3-5; relatively easy to very hard). When the number has been selected, the game begins.

You input your guess (the RETURN key is not needed –if a mistake is made, hit "R" to redo), then the machine "thinks" about your entry via changing colors and sounds generated by the input. It then prints its evaluation.

During the game, various prompts and sound effects are used; for example, if you haven't gotten the solution in 10 tries (on a 3-number game), the program displays the message, "You haven't got it YET?!" and gives you a "raspberry."

If the puzzle is solved, "Charge!" is played, along with the answer and number of tries, plus an evaluation of the game, e.g., "Not bad, but you need more practice!" Also shown is a "rating" system whereby you can see, for example, how many games you won in five tries. This display can handle double digits without getting physically larger. It also displays the total number of games played and the number of "give-ups." (If the puzzle is solved in a remarkably short time, the "give-up" count is reduced to zero for the rest of the game.) The game then asks for a replay. (Note: if the "number of digits" is changed, the "rating system" resets itself.) □

SEE PROGRAM LISTING ON PAGE 109

Mastermind is a trademark of Invicta Plastics.

PRINTAT

For the C-64

By Simon Edgeworth

This little program simulates the PRINTAT command of some versions of BASIC. It enables you to move the cursor anywhere on the screen with a single command.

The main program (lines 10-18) puts a short machine language program into memory, starting at location 679, and sets the variable AT equal to 679.

The command SYS AT, followed by two numbers, will now move the cursor to any desired location on the screen. For example: SYS AT,8,12: PRINT"MARS" will print "MARS" starting at line 8, column 12. SYSAT, L,C: PRINT"VENUS" will print "VENUS" starting at line L, column C. The demonstration program (lines 20-48) gives more examples of how the command can be used.

To use this utility in your own programs, you only need lines 10-18. Put them at the beginning of your program, and you will be able to use the new command for all of your screen formatting. □

SEE PROGRAM LISTING ON PAGE 111

FLOTSAM

Dear Mr. Allikas:

I have become a recent reader of *Ahoy!* I must say that you do an outstanding job as editor in bringing together a variety of informative and practical articles and features. Editorial matter is presented in an interesting manner with clarity and appeal to the eye.

You are to be commended for your many contributions in making *Ahoy!* a most valuable publication for the Commodore user.

Samuel Rabinoff, Principal
P.S. 207, Queens, NY

Ordinarily, we'd be too embarrassed to print such an overwhelmingly positive letter. But what editor could resist a letter from his elementary school principal?

I have been an avid reader of your magazine since your premier issue. I enjoy it very much and get a lot of useful information from it.

One objection I have that I feel should be brought to your attention are the ads from CVC Online that have appeared in the last few issues.

I am not a prude or moralist, but I do not think that a magazine of this quality is enhanced with advertisement of this nature. I am sure that the revenue obtained by accepting advertisement of this sort will be more than lost when circulation decreases due to those that do not care to have advertisement of that type put in the hands of their children.

I am taking the time to write, hopefully to encourage you to maintain the high quality of your magazine including the advertising that most of your subscribers read.

Thank you for your consideration of my observation and comments. I hope to remain a reader and advocate of your magazine.

John W. Carino
Columbia, SC



"Sure it's user-friendly! Who do you think bought this round?"

THE WIZARD OF IM

For the C-64

By Bob Spirko

In a remote corner of the galaxy, the planet Obid is suddenly assailed by a meteor storm. As this ancient planet hurtles through a swarm of rocks, its greatest city, called Im, is threatened with annihilation. A city council is hastily assembled and a decision unanimously struck: send for the Wizard of Im!

You are the Wizard of Im. Brandishing a magic wand that hurls huge fireballs, you set forth to destroy the deadly cascade of rock and spare the city. Despite your powers, however, it remains to be seen if you are quick enough to blast the multitude of menacing meteors before they fall to earth.

The Wizard of Im is a colorful action game that requires lightning reflexes. At first, you'll find it easy to demolish the meteors as they fall from the sky, but in time the speed and number of rocks increases so that you must work frantically to destroy them while at the same time keeping yourself from being struck down.

Since *The Wizard of Im* is written entirely in machine language, you'll need *Flankspeed* to type it in. Run *Flankspeed* (see page 86) and enter the following hexadecimal numbers:

> starting address: C000
> ending address: C71F

Once entered, save the program on disk or tape. Now reset your computer and type LOAD"WIZARD OF IM",8,1 for disk or LOAD"WIZARD OF IM",1,1 for tape. To start, enter SYS 49152.

The screen will display a brief introduction and then the action begins. Use your joystick to move the wizard left or right, and press the fire button to release a fireball. At the bottom of the screen, your score, lives, level, and best score are displayed. Each meteor that you vaporize will net you a few points or few dozen points, depending on the level that you are on. To your credit, the occasional meteor will disintegrate of its own accord before striking the ground. However, meteors that elude your fireballs and reach the earth will cost you points (except on the first level).

As you progress through the levels, the meteors appear more often and move faster. In time, the screen becomes so congested with hurtling rock that moving the wizard safely becomes difficult. You'll notice, too, that some levels are more formidable than others. This is because the value of each rock is increased with each level and then reset. This means that some levels will take longer to get through than others, but once through these the next few levels will be quicker.

Like all good wizards, you have three lives. Should you be struck by a meteor you will lose one life. Thrice killed, the action stops and you can start over by pressing any key. During the play itself you can reset the game by pressing f7, or you can exist back to BASIC with f8. To pause at any time press the SHIFT-LOCK key. □

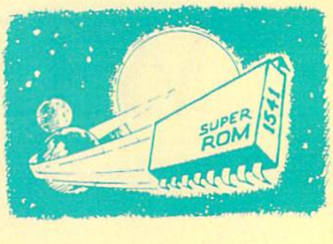
SEE PROGRAM LISTING ON PAGE 98

D-Compiler $59.95

The first D-Compiler to give you back your source code *after* your program has been compiled with **Blitz.*

*Blitz is a trademark of Skyles Electric Works

1541 Super ROM $39.95



- *Fast Save Load Verify*
- *Fast Scratch and Validate*
- *10 Second Format with Verify*
- *No more Drive Head rattling during Format or Error Reading*

also Save with Replace is Improved

- *Two times faster, Eight times faster when used with Turbo 64 which is included*
- *1541 Super ROM is 100% Compatible*
- *Easily installed in Minutes*

APALLO COPIES IT ALL



APALLO Does it All. This program is the latest generation of copy programs. It will do everything the $39.00 and $49.00 programs will do and **more.** It Copies **ALL** drive errors, bad tracks and sectors, non-standard format, bad syncs, and half tracks. We feel this is the best program of its kind available...

$29.95

MSD Sure Copy

At last a complete utility package for the MSD Dual Drive. ***This is the first MSD utility program that does it all.*** *The main menu options include:*

- *Copy Protected Disk*
- *Copy Files*
- *Format a Disk*
- *Change Disk Name*
- *Quit*
- *Copy Unprotected Disk*
- *Scratch a File*
- *Rename a File*
- *View Directory*

$39.95

Sure Copy *will put all errors automatically on disk: 20, 21, 23, 27 and 29's.*

D-CODER

- Translates any machine language program into easy-to-read English descriptions with complete explanations of each command!
- Makes complete notations of all important memory locations accessed by the program! (SID, VIC, MOS, KERNAL, etc.)
- Gives you three ways of accessing programs:
  1) **Will read and list programs from DISK**
  2) **Will read and list programs from MEMORY!**
  3) **Direct user input** (from magazines, etc.)
- Can be used to locate and examine any machine language program's protection routines!
- Can be used to easily break apart machine language programs for study and examination!
- Printer option for complete hard copy listings!

You no longer need to be an EGGHEAD to read Machine Language.



$19.95

N-CODER $19.95

the machine language manipulator...



The perfect companion program to D-CODER!

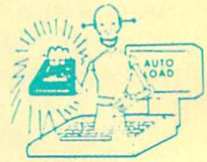
Allows you to easily make changes in machine language programs... right on the disk!

- *Rewrite ability allows code to be altered and then rewritten directly to the disk!*
- *Features sector-by-sector scrolling assembly language display of machine language programs!*
- *Notation of ASCII text equibalents for easy spotting of embedded text strings!*
- *Handy reference display of all assembly language commands and their ML numerical equivelents!*
- *Byte splitter for easy splitting of decimal addresses into low byte-high byte . mat!*

Super Loader

Super Loader is a Kartridge that plugs into your expansion port, that allows the computor, on power up start the disk drive and load the first preselected program on the disk.

- Change colors
- Load wedge
- Works with more drive
- Takes up no memory
- Reset switch included



Only **$29.95**

TOOL BOX

This Disk has over 100 routines, some of them are routines for protection, smooth scrooling, modem routines, and sound and color routines. They can easily be incorporated into all of your programs. It is also fully documented. **$19.95**

Slik Load

$29.95



Slik Load is a Kartridge for the C-64.
Slik Load is the most reliable, effective and thought out Kartridge of it's kind.
The options include:

- **5 times faster load**
- **Eliminates drive rattle when errors are encountered.**
- **Old and un-new Will restore a basic program**
- **Status key will give you information on device number, bytes free and status of the drive.**
- **Silk Load is also fully compatible with the 1541 Super Rom**



$19.95

DMS-Errors 20, 21, 22, 23, 27 & 29
Format Single Tracks
Read Disk Errors

½ Track Reader-read and select ½ track

½ Track Formatter-Format a disk with ½ tracks. This is where the next protection schemes are coming from.

Drive Mon-Disk Drive assembler/disassembler. For your 1541.

The Doc-Disk Doctor that reads code under errors.

Sync Maker-Place a sync mark on **any** track out to 41. Also used for protection.

Sync Reader-Check for Sync bits on **any** track out to 41.

Change Drive No.-Changes drive number (7-30)

Disk Logger-Finds starting track sector; start and end addresses.

Disk Match-Compare any two diskettes. Byte for byte.

New Wedge-Easier to use DOS wedge.

ID Check-Check ID's on **any** track.

Unscratch-Restore a scratched file.

View-BAM-Visual display of the free and used sectors on a diskette.

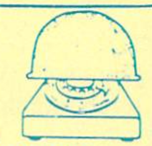
Read/Write Test-1541 performance test.

Repair a Track-Repair a track with checksum errors. Reads code under errors and restores track.

Fast Format-Format a disk in just 10 seconds (with verify!).

This is the only utility of its kind. It even has a 3 min. copy on it.

WAR GAMES AUTODIALER



1-**Auto Dial** will automatically dial a set of numbers you choose.
2-**Review Numbers** will review numbers that were answered by a computer.
3-**Save Numbers** will save numbers where a computer answered.
4-**Hardcopy of Numbers** will print out list of numbers where a computer answered.
5-**LOAD Numbers** will load in numbers to continue where it left off.
6-**Continue** will pick up dialing where it was interrupted.

$29.95

KARTRIDGE KRACKER

NOW you can own this unique and powerful tool which will allow you to dump the contents of 8K and 16K cartridges onto disk! But what's really great is that you can also RUN the cartridges programs without plugging in the cartridge! the KRACKER gets YOU INSIDE the cartridge! Put all your favorites on disk and get rid of the clutter. This package provides your with the software and hardware needed to get started. Program on disk included. (Some cartridges require use of external RAM not included)



$44.95

SOFTWARE PROTECTION HANDBOOK

3rd Edition

THIRD EDITION! NOW AVAILABLE!

If you're **tired** of being harassed by **protected software** and too many **copy** programs, then this is the book for you! This 224 page manual covers the gambit from legalities to protection methods to step-by-step back up procedures. Now you can learn both how to **protect and unprotect** software! The techniques covered include copying cartridges to tape or disk, tape protection, and disk protection. Disk protection covers error no.'s 20, 21, 22, 23, 27 and 29 **plus** single track formatting, header modification, header swapping, half track reading and writing, reading and writing modified bit densities, formatting illegal tracks/sectors, sync writing and **more!** The Third edition **explains,** tells how to **detect** and how to **write** them with included software. Eleven useful utilities and many protection listings! Our disk analysis programs reveal the protection methods used on your originals. A diskette with all software is available for a minimum extra charge. This may not be the only book your should have for the C-64, but it is certainly **the one** book you should not be **without!**

C64 Book only	$19.95 US
Book & Disk of all programs	$29.95 US
Vic 20 book . . . Cart. & Tapes only	$9.95 US

THIS MANUAL DOES NOT CONDONE PIRACY

*SHIPPING: $2.00

TOP SECRET STUFF II

All C128 Compatible

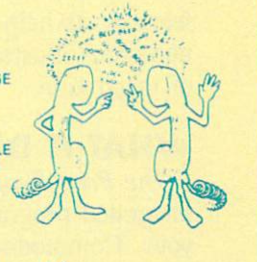
- Split Screens (Horizontally)
- Smooth Scrolling
- Save Ram From Under Roms
- No Drive Rattle On Errors
- Triple Drive Head Speed
- Autoboot Maker
- Koala Screen DUmp
- Display GCR
- Fast Disk Eraser
- Protect Scheme For Your Disks
- Write Protect Disk
- Unwrite Protect Disk
- Mini D.O.S. Wedge
- Fast Diskmatcher
- Data Statement Maker
- Unnew
- 3 Minute Copy
- D.M.S.

If you have TSS#1, You'll Like This One! All on one disk **$19.95**

Bulletin Board

Set up and operate your own bulletin board with one or two disk drives. This one has all the features and you can customize it easily yourself.

1-RUN MEGASOFT-BBS
2-CREATE MEGA FILES
3-ADD TO SYSOP'S CORNER
4-NEW SYSOP'S CORNER
5-READ MESSAGES
6-SCRATCH MESSAGE
7-CYCLE MESSAGES
8-READ SYSOP MESSAGES
9—WRITE OPENING MESSAGE
10-READ LOG
11-CYCLE LOG
12-READ DOWNLOAD FILE
13-SCRATCH DOWNLOAD FILE
14-ADD TO OTHER SYSTEMS
15-CREATE OTHER SYSTEMS



$59.95

BBS

- Leveled Access
- Private Message Base
- Up to 300 Passwords
- Expert Mode
- Open Chalk Board for Highest Level Access

Enclose Cashiers Check, Money Order, or Personal Check. Allow 14 days for delivery. 2 to 7 days for phone orders. Canada orders must be in U.S. Dollars. VISA — MASTER CARD — C.O.D.

Programs for C-64 — $2.00 S & H on all orders

Software Submissions Invited.

MegaSoft Limited

P.O. Box 1080, Battle Ground, Washington 98604

Phone (206) 687-5116 • BBS 687-5205 After Hours Computer-to Computer

PRINT SHOP

The Print Shop from Broderbund is a multipurpose graphic utility which turns the Commodore 64 with dot matrix printer into an all-purpose greeting card, banner, and poster



generator. It is one of the best thought out, easiest to use packages I've come across for the Commodore 64. It is so easy to use, in fact, that I was able to generate my first greeting cards without once referring to the manual. The manual, by the way, is a well-written, 28 page, 8½ x 11" softcover book. It does a thorough job of describing *The Print Shop*'s numerous features. To help get you started, the numerous illustrations are a source of inspiration.

WHAT IT DOES

The Print Shop is a collection of integrated programs that will allow your Commodore 64 to perform many of the services heretofore farmed out to your local printery. These consist of:

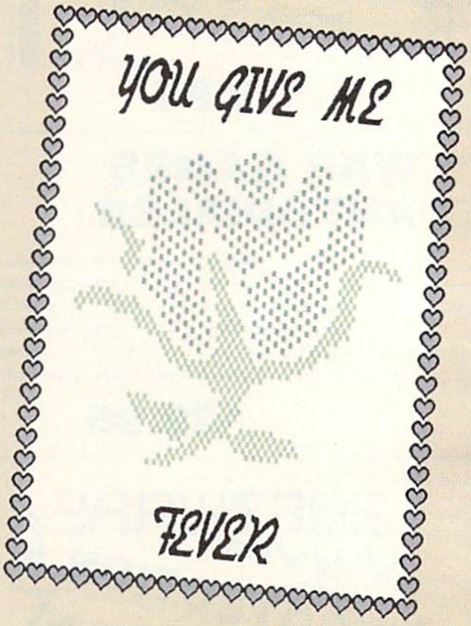
1. Greeting card generator—turns a sheet of paper into a quarter-folded missive, complete with messages on the front and inside as well as a credit line on the back (Hallmark, move over).
2. Sign printer—sure to be welcomed by many a small business, as it saves hours with a stencil. These single sheet displays have become all the rage at the local computer fairs and flea markets.
3. Letterhead generator—allows you to create fully illustrated customized stationery.
4. Banner printer.
5. Kaleidoscope generator, complete with bit map screen dump.

GREETING CARDS GALORE

The greeting card generator will most likely be the first part of this package you will try out. As with all the routines, a set of menus takes you through the entire process.

First you may choose one of nine possible borders or none at all. The borders range from simple lines through flowers and lattice work. Next you may choose a graphic image to dress up the card. Fifty pictures and ten patterns are provided on the program disk. The graphics may be selected by number off the handy reference card included with the package. If you cannot bear to tear your eyes from the screen, the program will prompt you with the names of all the prepackaged graphics. If none of the provided graphics suit your taste, a rudimentary graphics editor lets you design your own (more on this later).

If you do select a graphic, you are asked to choose from a small, medium, or large image. If you select one of the smaller sizes you will have the choice of both layout and number of images. Only one graphic may be used for a given part of the card. The graphics cannot be mixed on the same page of the card. The image may be used only once, or repeated up to 35 times to form a solid back-



ground. Different graphics may be selected for the front and the inside of the card. The large graphic may be printed only once in the center of the card.

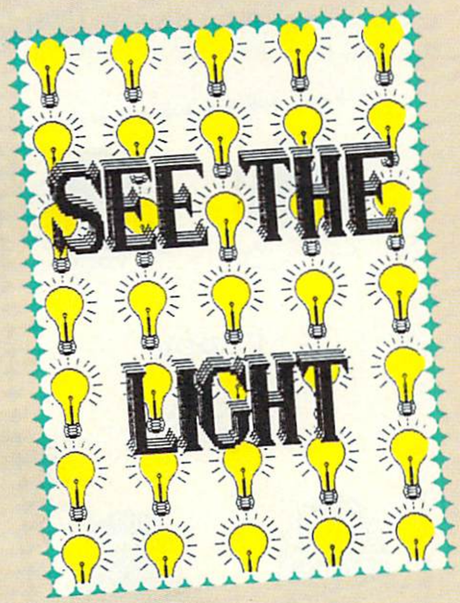
One of eight font styles are available for your message. These fonts range from an elaborate Alexia to a simple Block style. Individual text lines can be printed out in one of two sizes using solid, outline, or 3-D lettering with any of the fonts. Fonts cannot be mixed on the same page of the card, although different fonts may be selected for the inside and the front of the card. The more complex styles take up more space, which limits the size of the message. The amount of text available ranges from

The Print Shop ***cannot actually produce full color graphics like those shown here—Art Director JoAnn Case just could not resist coloring them in. All are reductions of dumps from various*** **Print Shop** ***utilities, including the greeting card generator, the sign printer, and (at bottom of page 51) the banner printer—the original of which is over five feet long.***

A detailed look at the popular C-64 program, plus two original enhancement utilities:
Print Shop to Bit Map Convertor by Morton Kevelson
Screen Magic to DOODLE! Convertor by Michael Beutjer

By Morton Kevelson

a maximum of 14 lines by 26 characters in the single sized Stencil font to a minimum of 4 lines by 8 characters in the double sized Alexia font. Individual lines may be centered or left or right justified.



The entire creation process is repeated for the inside of the card. The final step is an optional fifty six character credit line for the back of the card.

Actually, there is no need to follow the prompts in the order presented. The program allows for easy movement back and forth through the menus. The only thing to watch out for is the font selection process. If you scroll through any of the larger fonts or change the size of the text lines, any part of the message which no longer fits will be lost even if you reselect a smaller font.

PRINTOUT

The heart of the program is a very sophisticated print routine which is customized for your specific printer. Two versions of the program are provided. The back of the disk contains the Commodore 1525/MPS-801 version. The front of the disk supports a variety of popular printers including those made by Epson, Star Micronics, Panasonic, C-Itoh, Legend, NEC, Mannesman Tally, Blue Chip, and Okidata. A customizing routine modifies the original program disk, after which a write protect tab should be installed.

The entire 8½ x 11" surface is used for the greeting card. The paper may be precisely positioned by a test routine which advances the paper in 1/72" increments till a dotted line is printed precisely on the perforation. When printing is complete, you simply fold the paper twice to complete the card.

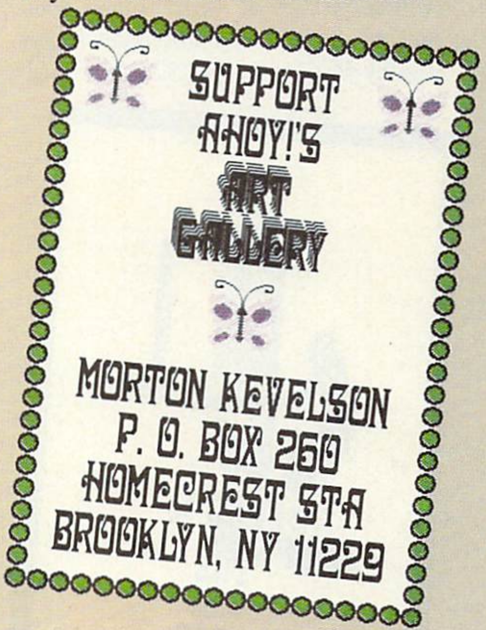
The printout process is rather long, about eight minutes with a Gemini 10X. A colorful animated display lets you know that all is well with the printer, program, and computer. The display message alternates between "thinking" and "printing" while the operation is taking place. Considerable disk access occurs during the printing process, as *The Print Shop* recalls data on the type fonts and graphic images.

SIGNS

The sign utility lets you create a full page sign for just about any announcement. The creation process is very similar to the above. Text space ranges from a maximum of 14 lines by 26 characters to a minimum of 4 lines by 8 characters, depending on font and size.

LETTERHEADS

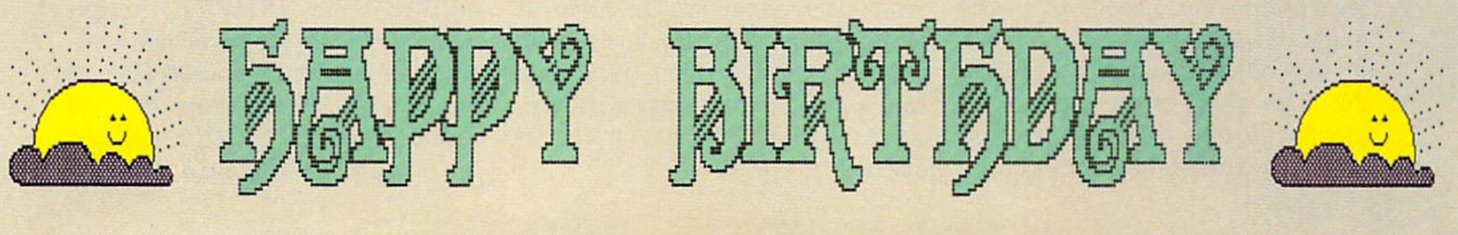
The letterhead utility lets you design customized stationery. Both a header and a footer can be created. Each consists of a 32 character line in your choice of font with three addi-



tional 63 character text lines. A horizontal separator line may be used to segregate the header or footer from the rest of the page. Graphics may be incorporated in the letterhead design.

BANNER

Need a long sign with eight inch high letters? *The Print Shop* banner utility fills the bill. There is no limit to the length of the message, as repeated applications are possible. This also allows for the mixing of fonts, as well as a variety of graphics on a single banner. Just keep a plentiful supply of printer ribbons and time if you plan to indulge. Roll paper is a definite plus for this application.



GRAPHIC EDITOR

For those finicky souls who cannot find a suitable graphic among the 60 with the package, the graphic editor lets you modify or create your own. This rudimentary utility is a far cry from the sophisticated packages I looked at in the October and November issues of *Ahoy!*, but it does the job. The 88 by 53 dot sketch pad (44 by 45 dots on the 1525/MPS-801



version) occupies about half the screen. Input is accepted from a touch tablet (best), a joystick or trackball (good), or the keyboard (fair). The main problem with keyboard entry is the lack of automatic repeat with the cursor keys.

The editor allows for the placement and removal of dots or clearing of the drawing screen. No other features, such as geometrics, fills, or block moves, are supported. The cursor coordinates are continuously displayed at the bottom of the screen. This will allow you to plot out the entire image on graph paper beforehand. Tracing a picture on a suitably scaled grid might be a good way to start. Graphics may be saved to disk for recall by the other parts of the package. A graphic file only occupies three disk blocks (two disk blocks for the 1525/



MPS-801 version). Thus up to 144 graphics, the capacity of the 1541 disk directory, may be stored on a disk.

PRINT SHOP TO BIT MAP CONVERTOR

The graphic editor has its own unique file format for storing the bit mapped image. The *Print Shop to Bit Map Convertor* program on page 88 of this issue will transfer a part of a Commodore 64 standard bit map file into the graphic editor format and vice versa. The program is set up to work with *DOODLE!* files and the non-Commodore *Print Shop* files. When going from *DOODLE!* to *Print Shop,* a small portion is extracted from the upper left hand corner of the *DOODLE!* image. In the opposite direction, the same size image is produced with the addition of a one pixel border. This will allow the extended features of the *DOODLE!* package to be used for the creation of *Print Shop* graphics.

SCREEN MAGIC

This is a unique high resolution bit mapped four quadrant kaleidoscope generator. Two basic patterns are generated. The first pattern has eleven possible variations. Once started, the images vary continuously in an almost hypnotic fashion. When you spot a particularly pleasing pattern, strike the RETURN key to freeze the display.

As with all the other *Print Shop* utilities, a message using any of the fonts may be superimposed on the bit map. Message capacity varies from 8 lines by 22 characters to 2 lines by 6 characters, depending on font and character size. The text may be modified without affecting the original underlying image for as long as you do not save or print the display. Thus, it would be a good idea to save the

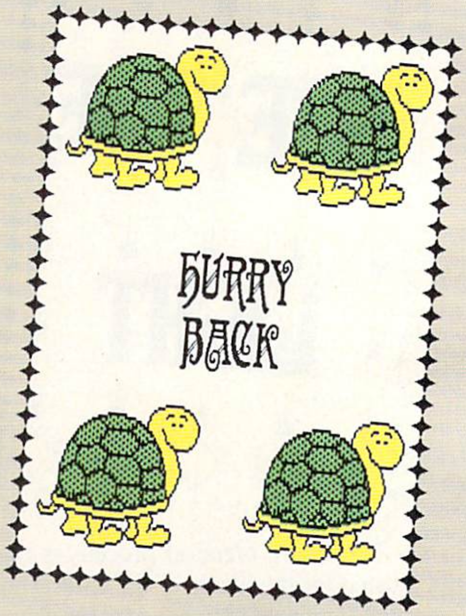


image to disk before experimenting with text. The resulting printout fills about a half of a page.

DOODLE! TO SCREEN MAGIC

According to the *Print Shop* manual, the *Screen Magic* images are not compatible with other Commodore graphic packages. Not any longer. Michael Beutjer, author of *Koala Printer,* has provided a program to convert between *Screen Magic* and *DOODLE!* format. This will allow you to color in or print out a *Screen Magic* image with the *DOODLE!* package. The *DOODLE!* screen dump yields an image which is either half the size or twice as large as the *Screen Magic* dump. See the *Screen Magic to DOODLE! Convertor* on page 88 of this issue.

THINGS I WOULD LIKE TO SEE

Several needed enhancements to the *Print Shop* package come to mind. The ability to save a greeting card design would be appreciated. Although the creation process is only a few minutes, being able to recall a saved card would avoid spelling errors and typos.

How about a mail merge utility? This would permit the lengthy printing process to be left as the last step after all your cards have been created. Convenient customization for that truly personal touch could be readily implemented.

A font editor would make a nice enhancement. If nothing else, this would allow the *Print Shop* to "learn" other languages.

Some musical accompaniment would turn the Kaleidoscope utility into a really effective display. Just imagine what the effect of a wall full of displays with the right sound would be like.

Last, but not least, a sophisticated graphic editor would be most appreciated.

CONCLUSIONS

The Print Shop is destined to become one of the most popular packages for the Commodore 64. Its ease of use and wide ranging utility will make it a must-have for most users. The suggested enhancements I have mentioned above in no way detract from the quality of the package as it is presently supplied. I predict that the popularity of this package will lead to many of these features, either directly from Broderbund or from outside sources. ☐

Broderbund Software, 17 Paul Drive, San Rafael, CA 94903 (phone: 415-479-1170)

SEE PROGRAM LISTINGS ON PAGE 88

LUCKY LOTTERY

For the C-64

In light of the money given away every week by the State Lottery Commission, I figured I could increase my chances of picking out the six numbers it would take to pick up the bucks if I had some way of keeping record of the numbers already chosen. Armed only with my trusty C-64 and programming skills, I set out to accomplish this task. After about five weeks of trial, error, and debugging, I finally came up with a finished program that will store all the past lottery numbers which have been chosen, display all possible numbers (depending on state) and the amount of times each has been chosen, and pick out the six numbers which have been most frequently chosen.

Now don't get me wrong, this program will not pick out the six numbers you need to win, but I figure the more information you have about something the more easily you can make an intelligent choice.

Operation of the program is fairly simple. Once past the title screen, you work from a main menu. When first starting your numbers collection, use the #2 selection "ENTER NEW NUMBERS". You will be prompted through the six digits required. The program will then display all 44 numbers. In the beginning a lot of zeroes will show on the screen, but this will change after a few weeks. You will then be asked to press "f1" for the six most common numbers selected. These are the numbers from which you can select your six choices. After the first time you use the program, you must select the first option "LOAD NUMBER FILE" to load the old numbers before you add your latest entries. One more note: always end your program with the #5 selection "END SESSION". This updates your file and ends the program.

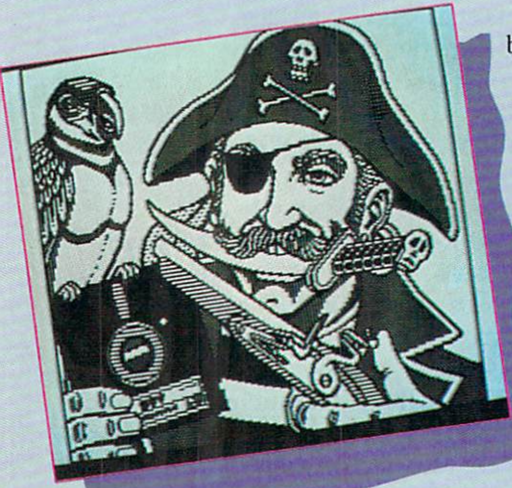
Lucky Lottery will only work with a disk drive. I hope you enjoy using it as much as I enjoyed programming it. And please, if any of you hit the lottery using my program, don't forget to cut me in. ☐

SEE PROGRAM LISTING ON PAGE 111

By Bob Lloret

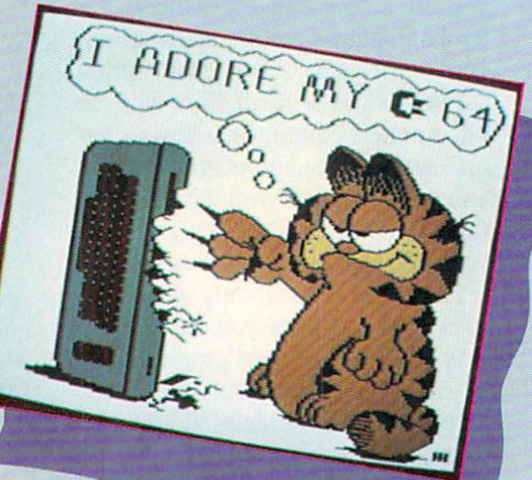
ART GALLERY





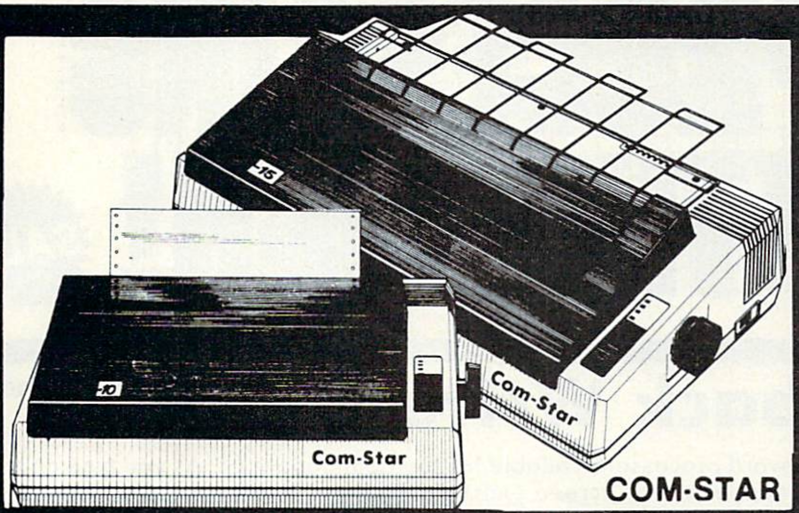


Clockwise from top: *Natalie* by Wayne Schmidt (New York, NY), created on *DOODLE!*; *Dragon* by Brian Glover (New York, NY), also on *DOODLE!*; front and back views of *Garfield* by Henry Ho, rendered on *Sorcerer's Apprentice*; and *Polly Wanna Crackit* by Wayne Schmidt, using *Flexidraw.* Contributors to *Ahoy!'s Art Gallery* will receive royalties based on the sale of disks containing the best computer graphics received, both published and unpublished. Send your best work on disk, accompanied by a stamped and self-addressed mailer, to Morton Kevelson, P.O. Box 260, Homecrest Station, Brooklyn, NY 11229. Indicate the drawing package that was used to create the image. If you employed a bit map of your own design, indicate the appropriate file parameters, i.e., hi-res or multicolor, location of bit map, screen and color data.





FANTASTIC COMPUTER PRINTER SALE!!!



10X COM-STAR*
HIGH SPEED
Tractor Friction Printer
130-150 CPS
Only
$199
List $499

15½" 150-170 CPS Printer $319⁰⁰

- *Lowest Sale Price, Premium Quality, Tractor/Friction Printer in the U.S.A. (Best Value)*
- *High Speed 130-150 Characters Per Second* • *40, 46, 66, 80, 96, 132 line spacing*
- *Word Processing, Letters* • *Business Forms* • *Labels, Graphics, Tables* • *List Programs*
- *Fantastic Graphics* • *Print Modem Data* • *The Most Important Accessory For Your Computer*

Premium Quality 130-150 CPS 10X COM-STAR High Speed Printer $199

10" carriage, 2K buffer, prints 8½"x11" standard single sheet or continuous feed paper, Bi-directional, impact, dot matrix, 130-150 CPS, 9 x 9 dot matrix with double strike capability for 18 x 18 dot matrix (near letter quality), high resolution bit image, underlining, downloadable characters, true lower descenders with super and subscripts, prints standard, block graphics, and special characters. It gives you print quality and features found on printers costing twice as much!! (Centronics Parallel Interface) (Better than Epson FX80 and Gemini 10x).
List $499.00. **Sale $199.00.**

Premium Quality 150-170 CPS 15½X COM-STAR Business Super High Speed Printer $319.00

Has all the features of the 10X COM-STAR PRINTER plus 15½" carriage and more powerful electronics components to handle large ledger business forms! (Better than Epson FX 100 & Delta 15).
List $599. **Sale $319.00.**

JUKI®

12" DAISY WHEEL PRINTER $199.00

"JUKI" Superb letter quality daisy wheel printer, 12" extra large carriage, up to 12CPS bi-directional printing, drop-in cassette ribbon, (90 day warranty) centronics parallel or RS232 serial port built in! (Specify).
Limited Quantities.
List $299.00. **Sale $199.00.**

JUKI®

Printer/Typewriter Combination $249.00

"JUKI" Superb letter quality, daisy wheel printer/typewriter combination. Two machines in one — just a flick of the switch. 12" extra large carriage, typewriter keyboard, automatic margin control and relocate key, drop in cassette ribbon! (90 day warranty) centronics parallel or RS232 serial port built in (Specify).
List $349.00. **Sale $249.00.**

Olympia

Executive Letter Quality $339.00
15" Daisy Wheel Printer

This is the world's finest daisy wheel printer. Fantastic letter quality, up to 20 CPS bi-directional, will handle 14.4" forms width! Has a 256 character print buffer, special print enhancements, built in tractor-feed (Centronics Parallel and RS232C Interface) (90 day warranty).
List $649.00. **Sale $339.00**

Olympia

Printer/Typewriter Combination $439.00

Better than IBM Selectric. Superb computer printer combined with the world's finest electronic typewriter. Two machines in one, just flick the switch for up to 20 CPS printing (300 **Words per minute**) on a 15" carriage that handles up to 14 1/8" in. paper. Drop in cassette ribbon — express lift off correction, Centronics parallel interface (90 day warranty).
List $749.00. **Sale $439.00.**

• *15 Day Free Trial — 1 Year Immediate Replacement Warranty*

PARALLEL INTERFACES

For VIC-20 and COM-64 — $59.00. Apple — $59.00. Atari — $59.00.

Add $14.50 for shipping, handling and insurance. Illinois residents please add 6% tax. Add $29.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. Canadian orders must be in U.S. dollars.
WE DO NOT EXPORT TO OTHER COUNTRIES, EXCEPT CANADA.
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days delivery. 2 to 7 days for phone orders. 1 day express mail!
VISA — MASTERCARD — C.O.D. No C.O.D. to Canada or APO-FPO

PROTECTO
We Love Our Customers
22292 N. Pepper Rd., Barrington, Illinois 60010
312/382-5244 to order

☆☆ NEW ☆☆ **Commodore - 64** ☆☆ NEW ☆☆

WORD PROCESSING BREAKTHROUGH!

SALE SALE

40 or 80 Columns in Color

Paperback Writer 64

40 or 80 Columns in Color

This is the easiest to use and most powerful word processor available for the Commodore 64. As you type on the screen, you will see your letters and words appear on the screen exactly as they will be printed (i.e. Italics will be *Italic*, Bold Face will be **Bold Face**). With the printer files you can customize Paperback Writer 64 to use all the fancy features of your printer. Loads EZ Script®, Paperclip®, & Wordpro 64® Files so you can easily upgrade your past wordprocessing text that you've written with obsolete wordprocessors.

Take a look at some of the other features:



- **Wordwrap** — No Words Break At The Edge Of The Screen.
- **Flexible Cursor Movement,** Including Tabs And Other Timesavers.
- **Deletion And Insertion** Of Characters, Lines And Blocks Of Text.
- **On-screen Text Enhancement**, Such As Bold Face, Italics, Underlining, Superscripts And Subscripts, And Foreign And Other Characters.
- **Manipulation Of Blocks (ranges)** Of Text For Functions Such As Moving And Deleting, Even Between Files.
- **Sorting Lists** In Order Of Numbers And Letters.
- **Aligning And Adding Numbers** In Columns, Helpful With Tables.
- **Variable Margins** At Left And Right, And Paragraph Indentation.
- **Lines Centered, Justified** Or **Aligned** At The Right Side.
- **Variable Page Lengths** And **Line Spacing.**
- **Borders** At Top Or Bottom With Optional Title Lines And Page Numbers.
- **Linked Files** To Print Extra-long Documents In Sequence.
- **Flexible Printer Set-up** To Allow Use With Any Printer.
- **Find And Replace** Text Functions That Can Be Automatic.
- Complete Or Selective **Directories** Of Files On The Disk.
- **Sequential Files** For Mail Lists And Communication With Other Computers.
- **Spelling Checker**, Checks Your Spelling.

INTRODUCTORY PRICE

$39.95

☆☆☆ Plus ☆☆☆

Full help screens on line with additional help on the disks mean you don't even need a manual. If you're in the middle of a page and you want to know how to use a special function just hit F7 and the information will appear before your eyes. If you still don't understand hit F7 again and a more detailed explanation appears. Then simply hit F8 and you're back in the letter where you left off. No manual lookup necessary. This is the easiest word processor in the world to use.

List $99.00. **Introductory Sale Price $39.95.**

DOUBLE PROTECTO FREE TRIAL

We are so sure this is the easiest and most powerful word processor available, we will give you 30 days to try it out. If you are not completely satisfied, we will refund your purchase price.

PAPERBACK DATABASE List $69.00. **Sale $34.95**. * *Coupon $24.95.*

Add $3.00 for shipping, handling and insurance. Illinois residents please add 6% tax. Add $6.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. Canadian orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA.
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail!
VISA — MASTER CARD — C.O.D.
No C.O.D. to Canada, APO-FPO.

PROTECTO

We Love Our Customers

22292 N. Pepper Rd., Barrington, Illinois 60010

312/382-5244 to order

EZ Script™ is a trademark of Commodore Business Machines. Paperclip™ is a trademark of Batteries Included, Inc. & Wordpro 64™ is a trademark of Professional Software, Inc.

Reader Service No. 153

SALE

80 CPS PRINTER

SALE

Smith Corona

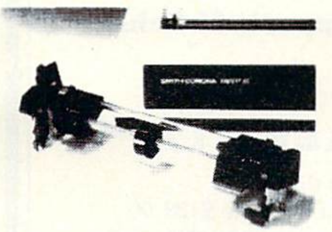
• Commodore 64 • Atari • Apple • More

• 11" Carriage • Friction/Tractor • Graphics

$159



1 Year Immediate Replacement Warranty



Tractor Feed Included FREE!

SMITH CORONA

The Fastext 80 dot matrix printer from Smith-Corona. A high speed, high flexibility printer with a low suggested manufacturer's retail price of only $299.00. **Sale $159.00**. It does your graphics with ease and prints letters with speeds up to **80 CPS**. ***(Does Commodore graphics with graphics interface.)*** As for flexibility, it has bi-directional printing, friction and tracter feed . Plus six different pitches including condensed and enlarged type for impeccable characters in a variety of sizes. Also standard is a full line buffer and thrifty drop-in ribbon cassettes that yield up to a million characters. Finally, it's compatible with virtually all personal computers. Teamed up with your personal computer, it'll keep track of your transactions, fly through finances, help with homework, even plan the family menu. All this with the assurance of superb Smith-Corona quality built in, makes the Fastext-80 a lot of dot for your dollar. List $299.00. **Sale $159.00.**

SPECIFICATIONS:

Size/Weight: Height 4" Width 16.5" Depth 9" Weight 8.2 lbs.
Electrical Needs: 120V/60Hz
Internal Char. Coding: ASCII & ISO
Print Buffer Size: 132 Bytes (1 line)
No. of Char. in Char. Set: 96 ASCII & International
Graphics Capability: Standard 60, 72 Dots Per Inch Horizontal, 72 Dots Per Inch Vertical
Pitch: 10, 12, 16.7, 5, 6, 8.3
Printing Method: Impact Dot Matrix
Char. Matrix Size: (Standard) 9H x 8V, (Elongate) 10H x 8V
Printing Features: Bi-directional, Short line seeking
Printing Speed: 80 CPS

PAPER

Type: Plain
Forms Type: Fanfold, Cut Sheet
Max Paper Width: 11"
Feeding Method: Friction Feed Std.; Tractor Feed Included

RIBBON

Type: Cassette — Fabric inked ribbon
Life: 1 million characters

CHARACTER MODE

Character Font: 9 x 8 Standard, 10 x 8 Elongated, No. 8 pin to be used for underline
Character Set: 96 ASCII, 11 x 7 International Char.
Pin Graph Mode: The incoming bit pattern corresponds to the 8 pins in the print head
Resolution: Horizontal: 60, 72 dots/inch, Vertical: 72 dots/inch

PARALLEL INTERFACES

Commodore 64 — $39.00 Apple II, IIe, II + — $59.00 Atari — $59.00

Add $10.00 for shipping, handling and insurance. Illinois residents please add 6% tax. Add $20.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. Canadian orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES, EXCEPT CANADA.
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail!
VISA — MASTER CARD — C.O.D.
No C.O.D. to Canada, APO-FPO.

PROTECTO

We Love Our Customers

22292 N. Pepper Rd., Barrington, Illinois 60010

312/382-5244 to order

Reader Service No. 154

COLOR MONITOR

Sale

SALE!!!

16 COLORS

(Premium Quality)

- **Built in Speaker and Audio**
- **Front Panel Controls**
- **For Video Recorders**
- **For Small Business Computers**
- **Apple - Commodore - Atari - Franklin - etc.**

Add $14.50 shipping, handling and insurance. Illinois residents please add 6% tax. We do not ship to foreign countries (including Canada), Alaska, Puerto Rico, APO-FPO.
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail!



13" Color Computer Monitor

(Premium Quality)

- **Beautiful Color Contrast**
- **High Resolution**
- **Separated Video**
- **Sharp Clear Text**
- **Anti-Glare Screen**
- **40 Columns x 24 Lines**
- **Supports 80 Columns**

List $399.00

SALE $179.00**

** with $19.95 Software Purchase
without Software Purchase $189.00
*Connecting cable $9.95

15 Day Free Trial - 90 Day Immediate Replacement Warranty

14" COMMODORE 1702 COLOR MONITOR

List $299.00 **SALE $179.00***

12" XTRON SUPER HI-RESOLUTION MONITOR

80 Columns x 24 Lines, Super Hi-Resolution 1000 lines Green or Amber Super-Clear "Easy to Read" text with special anti-glare screen!

List $249.00 **SALE $119.00***

12" ZENITH HI-RESOLUTION GREEN OR AMBER TEXT DISPLAY MONITOR

80 Columns x 24 Lines, Hi-Resolution, crisp clear easy to read text with anti-glare screen! A MUST for word processing.

List $199.00 **SALE $99.00***

12" MONITOR GREEN OR AMBER TEXT

80 Columns x 24 Lines, easy to read up front controls

List $159.00 **SALE $79.95***

* Connecting cable $9.95

• LOWEST PRICES • 15 DAY FREE TRIAL • 90 DAY FREE REPLACEMENT WARRANTY
• BEST SERVICE IN U.S.A. • ONE DAY EXPRESS MAIL • OVER 500 PROGRAMS • FREE CATALOGS

Add $10.00 for shipping, handling and insurance. Illinois residents please add 6% tax. Add $20.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. Canadian orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA.
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail!
VISA — MASTER CARD — C.O.D. No C.O.D. to Canada, APO-FPO

PROTECTO

We Love Our Customers

22292 N. Pepper Rd., Barrington, Illinois 60010

312/382-5244 to order

Auto-Append: An Easy Merging Utility for the VIC and 64

By William V. Braun

The thing that I least enjoy about programming is the time I must spend entering in code with the keyboard. I've often wished for some way of adding general subroutines that I had on tape or disk directly to the program I'm working on, rather than having to key them all in by hand. Then there were the times when I wanted to be able to load two programs into memory right from tape or disk and simply combine them into one program. Fortunately, I found a way to solve this problem. If you too have wished for an easy-to-use appending utility, I have the answer for you.

Auto-Append is a short BASIC program which loads a machine language routine into the top 28 bytes of RAM, and resets the limit of BASIC a bit in order to protect the routine from being overwritten. The machine language routine allows the user to enter two programs into RAM directly from tape or disk, then combines them into a single program. You can use *Auto-Append* with any amount of memory expansion, as the program automatically compensates for it.

Just follow these simple steps to append two programs:

1. Load and run *Auto-Append.*
2. Jot down the SYS locations in the instructions.
3. Load the first program (the one with the lower line numbers).
4. Execute the first SYS (type it and press return key).
5. Load the second program (the one with the higher line numbers).
6. Execute second SYS.

Step 2 is very important, as the instructions will gradually scroll off the screen as you load your programs and you will probably have forgotten the proper SYS locations.

It is essential that you keep in mind the fact that the first program you load *must* have lower line numbers than the second program. For example, if the first program ends with line number 800, the second program must begin with a line number no lower than 801. This is necessary due to the way programs are stored in RAM.

Once you have your programs combined, save a copy and verify it before trying to run it. A faulty POKE could cause the program to crash, making it necessary to repeat the appending process if you did not make a copy of the program. Also keep in mind that you must have enough memory (especially important on VIC 20's) to hold the machine language routine and both programs. This is unlikely to be a problem on the C-64; however, on an unexpanded VIC 20 you only have about 3.5K RAM to work with.

Adding your general subroutines (such as centering, rounding, etc.) to a program you are working on can be very easy if you store them on tape or disk as separate programs, then use *Auto-Append* to add them to your new program. You may want to give them meaningful names such as *Center, Round,* or *Hex Conversion.* Just load them from tape or disk as needed, using their program name. This technique should be much easier than having a single program with all the general subroutines in it, which would make it necessary for you to delete all the parts of it you did not want after it had been appended to your new program. Don't forget to give the different routines different line numbers, and make sure they are very high numbers to reduce the possibility of having them lower than the highest line number in your new program.

Let's take a look at the program itself. Line 10 is really the workhorse of the program. The first two POKEs lower the limit of BASIC by thirty bytes, protecting the machine language routine. Memory locations 55 and 56 are pointers for the end of BASIC. The variable A is the address where the first byte of the machine language routine is to be located. It is also the first SYS address. The FOR-NEXT loop in Line 10 actually reads the machine language routine which is held in the data statements and POKEs it into memory.

Lines 15 and 20 clear the screen and print the instructions on how to use *Auto-Append.* The second SYS address is equal to A+19. Line 25 holds the machine language routine executed by the first SYS, while Line 30 holds the second routine.

The first machine language routine moves the start of BASIC to the end of the first program you have loaded, thus causing the second program loaded to be located in memory right after the first. The second machine language routine returns the start of BASIC to its original location. You will notice that if you try a LIST after you have executed the first SYS it will appear that your first program is no longer in memory. This is just because the start of BASIC is now after that program, and a LIST causes the computer to start looking for a program where there is none. The second SYS returns the start of BASIC pointers to their original value, allowing you to see both programs as one. □ *SEE PROGRAM LISTING ON PAGE 100*

REVIEWS

THE GAMES CREATOR
Mastertronic International
Commodore 64
Disk; $19.95

Calling all would-be computer game designers. If you've got the ultimate action challenge bottled up inside you, but don't have the know-how to get the idea onto a floppy disk, David and Richard Darling's *The Games Creator* is a "must buy."

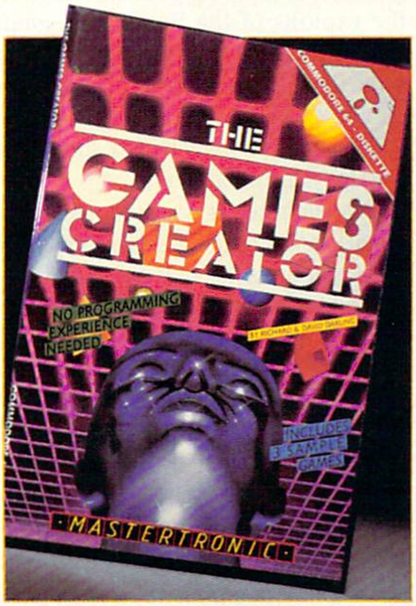
Software which lets non-programmers create adventure games and branching stories has become quite common in the last year. Until now, no comparable system has existed for action contests. (Broderbund's *The Arcade Machine*, released in 1983, is easier to use than assembly language or BASIC, but its process is still pretty complex.)

Any reasonably intelligent person can use this program to produce a climbing-and-jumping contest, maze, or scrolling shoot-out with extensive sound and graphics. A sample of each format is included on the disk, and most novices will start by modifying one of these sample games before constructing a new one from scratch. They're also reasonably entertaining in their own right, a bonus for consumers.

The 36-page instruction book reduces the gamemaking process to a series of small, easily understandable steps. The numerous illustrations clarify every phase of the process, even though fuzzy reproduction and sloppy photo cropping undermine the pamphlet's aesthetics. A glossary acquaints users with any unfamiliar design terms.

An onscreen menu, operated with a joystick or the keyboard cursor controls, is the hub of the system. Selecting "Change," as opposed to "Play," "Save," or "Load," begins the construction of a game.

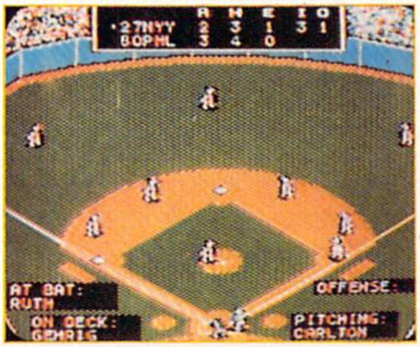
The designer can work on aliens, sounds, graphics, or the player. When one aspect of the game is done, the f7 key restores the main menu so the gamer can choose another.



Must buy for frustrated programmers.
READER SERVICE NO. 101

Sub-menus regulate the aliens' onscreen motion and behavior characteristics. There are five aliens from which to choose. The inventor can adjust random movement parameters, use the joystick or cursor keys to outline a predetermined movement path, and select the alien's starting position on the screen. The other menu controls animation speed, movement speed, the ability to scroll off the screen, shooting, the mechanism for advancing to higher levels, collisions, and the number of nasties.

Sound is divided into tunes and effects. A serviceable music construction kit can be used to write up to seven pages of music and make it sound like any of several instruments when played. A program can employ up to seven sound effects to aurally indicate when the player fires a bullet, gets killed, collides with an alien, and so forth. The individual effects can be styled by adjusting the attack, decay, frequency, and waveform. The computerist can check a sound by hitting f1 while working with the effects menu.



Manage the greatest teams of all time.
READER SERVICE NO. 102

Two sub-menus split graphics into scenery and sprites. The former permits the pixel-by-pixel construction of playfields; the latter focuses on the game's moving objects, or sprites, of which there can be up to eight.

The final menu commands the computerist's own onscreen representative. Among the user-determined factors are the player's starting point in the game, the keys which cause movement, how far a character can fall in a jumping game, and whether the player moves continuously or only in response to specific directives.

The obvious question is: What about the resulting games? *Games Creator* programs aren't state-of-the-art stunners, but they can be attractive and a lot of fun to play. The Darling Brothers have streamlined the construction process by restricting choices, especially in sound and graphics, without leeching the life out of it. With practice, an imaginative computerist can invent action programs which would be a credit even to the code-wizards who developed *The Games Creator.*

Mastertronic International, 407 Park Avenue South, Suite 16A, New York, NY 10016 (phone: 212-213-0166). ***—Arnie Katz***

MICRO LEAGUE BASEBALL
Micro League Sports Association
Commodore 64
Disk; $39.95

Strike three! The ballgame was over. And even before the Mighty

Casey finished his post-game shower, Mudville rooters were second-guessing the slugger's manager over a round of brews. One secret of baseball's enduring popularity is its endlessly fascinating strategy.

Statistical simulations like *Micro League Baseball* give armchair managers the chance to improve on reality. Their mathematical formulas quantify the performance of major leaguers so that each athlete in the game reflects the real-life player's abilities during the season on which the program is based. So if you replayed a full season, using each player exactly as his flesh-and-blood manager did, team and individual statistics would virtually duplicate the actual season's totals.

While some players of statistical simulations do, in fact, try to precisely recreate the past, most want to change history. Could a revised pitching rotation improve a team's finish? Is there someone riding the bench who ought to be playing every day? Would the team score more runs if it emphasized the hit-and-run over longball power?

Micro League Baseball gives computerists the chance to answer these and other equally intriguing questions. This is a rock-solid stat game dressed up in visuals which would do any action baseball program proud.

One or two human managers—there's a robot pilot for solitaire fun—can choose any of the 25 famous teams on the Game Disk. The publisher plans Team Disks, available for purchase separately, which will contain many more squads, including all-time all-star rosters for each major league franchise and disks that simulate full seasons. An easy data transfer routine lets the computerist match a team on any *Micro League Baseball* disk with any other team on a similar disk. Thus, it's possible to see how the Yankees of Mantle and Maris might stack up against the Bronx Bombers of Ruth and Gehrig.

The Game Disk contains an anthology of teams, intended to show the scope of the manufacturer's intended future releases. The Game Disk includes some recent juggernauts ('78 Yankees, '80 Phillies, '80 Royals), classic teams of the past ('27 Yankees, '55 Dodgers), and some all-star squads (all-time AL and NL greats). And masochists will thrill to the exploits of the numbingly inept '55 Washington Senators, one reason why there's no team in the nation's capital today.

Each team comes with a suggested lineup. Onscreen menus help a manager switch things around to suit personal preference. The program provides mounds of statistics to aid in the selection process. Won-lost record, games saved, ERA, games pitched, complete games, innings pitched, hits allowed, strikeouts, and walks are listed for all pitchers. The rest of the team members are rated for batting average, home runs, runs batted in, times at bat, hits, doubles, triples, stolen bases, and field average for any positions they can patrol on the field.

The defensive pilot chooses the hurler's delivery from among four possibilities and positions the infielders. They can also call for a pitch-out, order an intentional walk, or visit the mound.

The offensive manager can have the batsman swing away or lay down a surprise bunt. The skipper directs traffic on the basepaths and can order a sacrifice or hit and run.

Players view the action from a seat in the upper deck behind home plate. Small windows indicate who is at bat, on the mound, and in the on-deck circle, while the centerfield scoreboard prints pithy comments as the game progresses. Although the players are drawn small on the screen, wonderfully intricate animation makes their movements easy enough to follow.

Sound effects and music are more than adequate, but they slow down play. Computerists can cut playing time to about 30 minutes by hitting "M" when prompted. This clips the lengthy musical interludes and the admittedly impressive spectacle of having the ballplayers run on and off the field after each half-inning.

Micro League Baseball's flaws are few, but still worth mentioning. The inaugural edition doesn't print a box score or allow teams to swap players. The company has released a Utility Disk, priced at $19.95, which adds these functions.

The pitching system has a few kinks. Each throw merely symbolizes one complete interaction between a hitter and pitcher, so it's not realistic to call the hypothetical toss a "curve" or "fastball." Also, many pre-World War II pitchers didn't throw a slider. Perhaps this option on the pitching chart could be made more flexible to allow for a "specialty" pitch like a forkball or knuckler.

Documentation is outstanding. There is a rule book, profiles of all clubs on the Game Disk, and two cards which summarize the keyboard-based order-entry system.

Micro League Baseball is highly recommended for baseball-loving computerists. It provides plenty of mental stimulation and authentic action in an attractive package.

Micro League Sports Association, 28 E. Cleveland Avenue, Newark, DE 19711 (phone: 800-PLAYBAL).

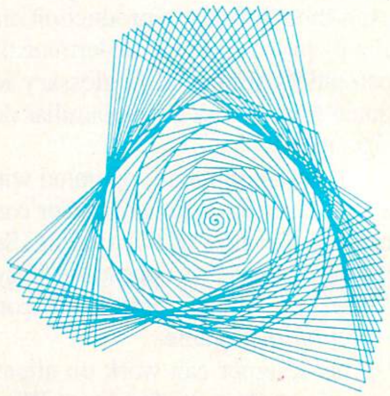
—Arnie Katz

PENMAN ROBOT PLOTTER

Axiom Corporation
Commodore 64
$399.00

When I discovered that the Penman plotter had arrived for review, I was ecstatic. This plotter appeared to be one of the most exciting peripherals to come along in a while.

Traditional plotters have a pen that moves about over a piece of paper by means of a rail in each direction (x and y). More recently, plotters have



Reduced-size version of Penman plot.
READER SERVICE NO. 103



We're sure you're looking forward to trying out the programs in this issue of *Ahoy!* But we're equally sure that you're *not* looking forward to typing them in. If you're an average typist, that should take you upwards of 25 hours. Not counting time spent correcting your typing errors, of course. How long that will take is anyone's guess. • How would you like someone to type the programs for you? At a price of, say—32¢ an hour? Wouldn't it be worth 32¢ an hour to free yourself up for more pleasant pursuits—like enjoying the rest of your *Ahoy!* magazine? • If you order the *Ahoy!* Program Disk or Cassette, you'll be getting that service for just under 32¢ an hour. Because for $7.95 (postage included)*, we'll mail you all the programs in this issue, on a disk or cassette that's tested and ready to run with your C-64 or VIC 20. If you subscribe, you'll save even more: 12 monthly disks or cassettes for $79.95, 24 for $149.95 (postage included)*. Back issues are available as well. • You already know how to type. Why not use the hours you spend with your Commodore to learn something new?

*Canadian and foreign prices higher.

Ahoy! *uses only*
maxell
premium quality disks.

SINGLE ISSUE AND SUBSCRIPTION PRICES

(Postage and Handling Included)	U.S.A.	CANADA	ELSEWHERE
❑ July '85 *Ahoy!* Disk ❑ July '85 *Ahoy!* Cassette	$ 7.95	$ 10.00	$ 12.00
❑ 12-Month Disk Subscription ❑ 12-Month Cassette Subscription	$ 79.95	$ 99.95	$124.95
❑ 24-Month Disk Subscription ❑ 24-Month Cassette Subscription	$149.95	$179.95	–
❑ *Best of '84* (disk only)	$ 10.95	$ 10.95	$ 10.95

BACK ISSUES $8.95

(except months listed)

In Canada add $2.00 per issue; outside US and Canada add $4.00 per issue.

❑ Jan. '84	$9.95	❑ ______	$8.95
❑ ______	$8.95	❑ ______	$8.95
❑ ______	$8.95	❑ ______	$8.95
❑ ______	$8.95	❑ ______	$8.95

SPECIFY
❑ ***DISK*** ❑ ***CASSETTE***

Make check or money order payable to **Double L Software.** N.Y. State Residents must add 8¼% sales tax.

Send coupon or facsimile to:
***Ahoy!* Program Disk/Cassette**
Ion International Inc.
45 West 34th Street, Suite 407
New York, NY 10001

NAME________________________
ADDRESS________________________
CITY______________ **STATE**______ **ZIP**__________

READER SERVICE INDEX

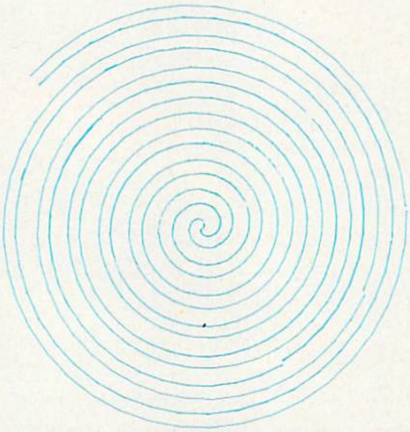
Page No.	Company	Svc. No.
22	APCAD	112
11	Abacus Software	148
C-4	Access Software	145
26	Ace Software	114
14	Anchor Automation	138
14	Anchor Automation	139
14	Apricorn	141
38	Artworx Software	170
64	Axiom Corporation	103
21	Big Bytes	158
15	Brother International Corp.	143
15	Brother International Corp.	144
79	CVC Online	111
C-2	Cardco, Inc.	122
8	CompuServe	115
47	Computer Management Corp.	108
41	Computer Warehouse	149
16	Custom Programming Group	159
23	Electronic Arts	124
5	The Ennon Corporation	117
35	Entrepo, Inc.	116
72	Epson America, Inc.	134
69	Full Circle Software, Inc.	106
25	Genesis Computer Corp.	160
9	Hayden Software Co., Inc.	126
12	Howard W. Sams & Co., Inc.	135
12	Howard W. Sams & Co., Inc.	136
114	Inexsoft	174
6	Innovative Organizers	150
67	Integrated-Software Systems	107
74	Intelligent Software	167
13	Island Digital Concepts	168
74	Jason-Ranheim	163
9	Mastertronic International	127
63	Mastertronic International	101
48-49	MegaSoft Limited	147
63	Micro League Sports Assn.	102
75	Microvations	164
4	Micro-W. Distributing, Inc.	169
45	Micro-W. Distributing, Inc.	104
17	N-Systems	119
20	Navarone Industries, Inc.	172
44	Ohio Computer Services	109
7	Okidata	125
14	Omnitronix	140
40	Parsec Research	120
10	Practcorp International, Inc.	130
36	Professor Jones	121
15	Progressive Peripherals & Software	142
56-61	Protecto Enterprizes	151-156
76	R.J. Brachman Associates, Inc.	110
55	Radix Marketing	157
62	Robbins Software	171
46	Roger Wagner Publishing, Inc.	113
71	SKOK Software	132
83	Sage Software	118
24	Solid State Software	162
12	Star Micronics, Inc.	137
C-3	SubLOGIC Corporation	146
68	T&D Subscription Software	105
10	Talktronics, Inc.	129
70	Ultrabyte	131
12	VMC Software	123
46	Wedgwood Rental	165
72	White House Computers	173
10	Woodbury Software	128
114	Xetec, Inc.	133

The publisher cannot assume responsibility for errors in the above listing.

sported a pen that moves only in the x direction, while the paper moves to provide y-axis movement. The Penman is completely different. It features an intelligent base unit connected to a "robot" via a ribbon cable. This robot holds three pens and plots by rolling around on top of the fixed plotting surface.

The Penman is ingeniously engineered. It has two wheels that drive it, and one unrestricted wheel for balance. When initialized, the Penman aligns itself with the edges of the paper by rolling toward the edge until its bottom optical sensors detect a contrast difference between the paper and background; then it backs up, turns, and seeks the other edge. All this is purely fascinating to watch.

The Penman can be told to plot in



A reduction of another **Penman** *plot.*

two ways: by Cartesian coordinates or polar coordinates. Plotting each way is very simple. The plotter is simply given a set of coordinates, or a direction and distance. Smooth circles are drawn in polar mode by giving a curvature parameter. An additional robotics mode gives you direct access to the drive motors and optical sensors for more direct control of the robot unit. A standard character set is also built into the plotter. Text can be plotted at various sizes and angles with ease.

Interestingly enough, all lines, circles, and shapes drawn by the Penman are completely smooth, in contrast to the stepped lines drawn by other inexpensive plotters. This is due to the fact that the Penman draws by continuously varying the speed of its two drive motors to produce smooth plots. Another interesting feature of the Penman is its ability to "unwrap" itself automatically from the connecting cable.

The only gripe I have with the Penman is that on complex plots accuracy is lost. One way around this is to have the robot realign itself every so many moves, but this takes time.

The Penman interfaces through any RS232 port and is driven by standard ASCII commands. At a price of $399, it is a surprising value in a low cost plotter.

Axiom Corporation, 1014 Griswold Avenue, San Fernando, CA 91340 (phone: 818-365-9521).

—David Barron

VISION BBS V8.3

Vision Software Co.
Commodore 64
Disk; $69.95

As the System Operator (SYSOP) of this magazine's bulletin board service (BBS) and longtime SYSOP of my own BBS, I have spent a great deal of time examining the BBS software currently available for the Commodore 64. With the wide variety of BBS's now in use, choosing the software that offers the best value for your money is not simply a matter of buying the most expensive you can find. Fortunately, Vision Software has recently released an outstanding entry into the marketplace.

Vision BBS V8.3 offers all the features you would expect of a good BBS, combined with a price tag that won't destroy your budget. With features such as new style Punter up and downloads, up to 99 different message bases, and a userlog that can hold up to 999 users, plus the ability to work with as many as five drives, this software is a joy to use. Though the clear, well-written documentation claims a setup time of 1 hour, those of you who are new to bulletin boards should allow a little extra time to plan out the many different menus that must be written.

Once the various menu files have been created, the BBS is ready to go online. The first user entered should naturally be the SYSOP, and subse-

quent users are added to the userlog as they call in. Each user is required to choose an 8-digit password and is assigned a user code as well. New users may be given a low access level until the SYSOP has time to examine their application, whereupon their security level can be raised to allow them access to more features on the BBS. Security levels can range from a low of 1 to a high of 99.

One especially useful feature of this software is the ability of the SYSOP, and selected users, to control the operation of the BBS from a remote terminal. By selective use of high access levels, the SYSOP can allow various users to log on and control individual areas of the BBS from their homes. This reduces the amount of time the SYSOP must spend maintaining the BBS and updating the files.

Another very interesting function of this program is that it can also be used as a terminal program, allowing the SYSOP to log on to other BBS's to trade files and information without the need to load in a different program. This feature alone is, in my opinion, worth half the cost of the software.

One drawback that I noted while setting up my preliminary copy of the BBS is the lack of a text editor, which is required to generate the various menus used. A quick call to the manufacturer brought swift assurances that all future versions will include not only a text editor but a timing system that will remove the BBS from the phone line at a predetermined time if you choose to run your BBS on limited hours.

Persons interested in viewing this software before buying may call the *Ahoy!* BBS at (718) 383-8909, 24 hours a day, 7 days a week.

The current price for the software is $69.95, and registered owners can purchase a backup copy for an additional $10.00. Future updates will also be made available to registered owners at a cost of $25.00. More information may be obtained by calling the Docs 'R' Us BBS at (914) 668-3664, or by writing the manufacturer at the address given below.

When ordering, please specify which type of modem you plan to use. There is a $10.00 charge to change modem types at a later date.

Vision Software Co., P.O. Box 534, Bronx, NY 10461 (phone: 212-829-1538 voice; 914-668-3664 data).

—B.W. Behling

SOUNDCHASER 64

Passport Designs, Inc.
Commodore 64
$199.00

At first glance, the Soundchaser 64 musical keyboard appears to be first rate. Unfortunately, the supplied software doesn't enable the keyboard to live up to the expectations set by its appearance.

To start on the positive side, the keyboard is a full-sized 4-octave unit with an excellent feel. The housing is attractive tan plastic, matching the Commodore. Connection to the computer is made via a ribbon cable to the cartridge port. Overall, this is one of the finest keyboards I have encountered.

The *BASIC Synthesizer Software* provided includes a monophonic and a polyphonic synthesizer. The monophonic package is reasonably good; it allows all oscillators to play in unison, producing a very 'fat' sound. Additionally, there is a LFO (low frequency oscillator) that can be used to produce vibrato and other special effects. Thirteen presets, which can be modified at will, are included. They are good, but I was able to improve on most of them through the included sound editor.

The polyphonic synthesizer allows you to play the keyboard with up to three notes pressed at once. Six presets are included; they are fair, none being very impressive. The limitation on the presets is caused by the lack of any special features as found on the monophonic package.

Neither package makes any provision to 'record' in any way. Modified presets cannot be saved to disk, and the mono/poly programs are separate and not directly accessible from one another. Fortunately, Passport told us that they will release versions of *Macmusic* and *Computer Sheet Music* that will take advantage of the

Soundchaser. Hopefully, these will make an improvement in a device with great potential.

Passport Designs, Inc., 625 Miramontes Street, Suite 103, Half Moon Bay, CA 94019 (phone: 415-726-0280). *—David Barron*

VIZASTAR INFORMATION PROCESSOR

Solid State Software
Commodore 64
Cartridge and disk; $119.97 or $149.97 (see below)

"Innovative," "awesomely powerful," "sorely needed," and "incredible guarantee" are phrases I immediately associate with *VizaStar.* These words aptly summarize this unique "work processor." Other apropos descriptors: "humongous cell capacity," "impressive graphics," "multiple windowing," and "easy-to-use;" the list goes on and on, but you have the idea: this is a program unlike anything else for the C-64!

VizaStar is an integrated software package; that is, it consists of three programs that work interactively. Residing simultaneously in memory, the spreadsheet, database, and business graphics programs are always immediately available. Each is outstanding in its own right. By integrating their features, Solid State Software has provided tremendous capabilities for information handling on the C-64. The only comparable product would be *Lotus 1-2-3* for the IBM PC; nothing in the C-64 world comes close to the vertical integration features of *VizaStar.*

The terms information processor and work processor, used by Solid State, describe not only what *VizaStar* is, but what it does.

All functions and applications for the program are menu-driven, and selections are supported by sub-menus. The basic display consists of a worksheet grid, and all labels, data, formulae, etc. are entered on it. Multiple windowing capability (up to 9 windows "open" at a time) allows viewing several portions of the worksheet simultaneously for comparison or reference purposes.

The program will support most popular printers suitably interfaced. Although a printer isn't mandatory to use *VizaStar,* it certainly is desirable. As of this writing (4/25), only the 1541 Disk Drive is supported.

The spreadsheet grid is a matrix of 1000 rows and 64 columns, yielding a maximum capacity of 64,000 cells! Cell formatting either individually or globally is provided in currency, scientific, date, left- or right-justified, centered, integer, and general purpose formats. Additionally, they can be globally or individually protected from modification.

Built-in functions for use in formulae include date processing and date calculations, in addition to the normal + – * / mathematical operators. The date functions are extremely useful and can be used for schedule planning, invoicing, "time-stamping" data, and multitudes of other applications. The built-in functions consist of:

CALENDAR: ©date, @day, @month, @year, @today

LOGICAL: @false, @true, @isna, @iserr, @if, @err, @na, and, or, not

MATHEMATICAL: @abs, @cos, @exp, @int, @log, @round, @sin, @sqrt, @tan, @pi

STATISTICAL: @avg, @count, @max, @min, @sum

TABLE LOOK-UP: @hlookup, @vlookup

A very powerful feature of *VizaStar* is the automatic keyboard "EXEC" facility. This allows you to set up a sequence of commands which can be executed at any time, further expanding the program's range of complex applications.

Database operations are called up from the worksheet, and a collection of up to 15 indexed files is possible. Each file holds records which can be up to 8,000 characters in length. The records may contain up to 64 different fields, and the fields can be up to 120 characters long. The average disk access time for any random record is three seconds on the 1541, fast enough to be envied by some "dedicated" database programs.

Database file layouts are easily configured on a blank screen and reversed headings and borders are possible. You can include up to nine screens per file layout, more than sufficient for just about any application.

All database processing is direct in the program, and search criteria may be combined if desired. Search operators include:

*—wild card match

?—individual character position match

&—match if following characters are found

< —less than match

> —greater than match

< ——ignore/do not match

Business graphics functions allow translating spreadsheet/database information into line and bar graphs, and windowing is permitted and supported in this mode as well. Additionally, the XGP (Extended Graphics Package) is included with the program, which provides two impressive and sophisticated graph modes—color "pie charts" and 3-D, 4-way-scrolling "skyscraper" graphs. These graphing formats are the best I've seen yet for the C-64, bar none.

The XGP graphs (pie & skyscraper) may be labeled and printed out directly on a Commodore MPS 801 or

Epson-compatible printer with suitable interface. Line and bar graphs are supported on almost all popular printers suitably interfaced. The printer functions will also support printing reports and labels.

VizaStar is extremely fast in terms of records access speed and overall operation due to the fact that it is written in 100% 6502 Machine Language. It is compatible with most word processors for the 64.

The *VizaStar* software resides both on cartridge and disk; the cartridge must be inserted while using the program. Two copies of the program disk are provided. In addition, a Tutorial Guide manual and a spiral-bound User Reference Guide manual are included, all inside a partitioned vinyl case that provides convenient and durable storage. Rarely do you find documentation equal to the caliber of the software it pertains to. *VizaStar* gets a solid 4-star rating in this department.

VizaStar has a retail price of $119.97 (version XL4; see below), not expensive when you consider that a good database will cost you about $100, as will a spreadsheet with decent graph capability, but not a small piece of change, either. Solid State Software is so confident in the product that they offer a full refund for any reason by returning it to the place of purchase within 15 days. After using *VizaStar,* I predict that Solid State will have very few refunds to make.

Though it's an incredibly powerful piece of integrated software, I found *VizaStar* quite simple to use, thanks to the excellent tutorial and reference manuals. The onscreen menus and sub-menus present the program options and functions in a clear, logical manner. Learning to use *VizaStar* is painlessly accomplished by following the tutorial guide and entering the examples. Unique features like the EXEC facility make it possible for novices to set up and execute complex command sequences.

Owing to the far-ranging capabilities of the program, it can be used for all sorts of applications. I found it particularly well suited for planning, projecting, visual analysis of data, scheduling, maintaining customer records, mailing lists, generating reports, and other information-handling tasks. I'm sure that the useful applications of *VizaStar* are limited only by the user's imagination.

If you're looking for a serious application program to handle your spreadsheet, database, and business graphics tasks, your search has ended. *VizaStar* does it all, and does it all amazingly well.

(*VizaStar* is available in two versions: the standard XL4 with 10K of free RAM ($119.97) and the XL8 with 14K of free RAM ($149.97).)

Solid State Software, 1125 E. Hillsdale Blvd., Suite 104, Foster City, CA 94404 (phone: 415-341-5606).

–Tom Benford

ON STAGE ($24.95)
MUSIC VIDEO HITS ($24.95)
3001 SOUND ODYSSEY ($39.95)
Sight & Sound Music Software
Commodore 64
Disk

"Just let me hear some of that rock & roll music, any old way you choose it."

Any old way? Chuck Berry might have had a few second thoughts if he had anticipated the invention of the home computer. Thirty years after those words were written, rock and roll has transcended the world of electric guitars and drums and entered to world of microprocessors. One of the first examples of this phenomenon to reach the marketplace is Sight & Sound's new "computer song albums."

It was only a matter of time before somebody figured—"Kids love rock, kids love computers, the C-64 has tremendous musical capabilities, so let's turn some rock songs into computer-generated music."

That's exactly what they've done. *On Stage* and *Music Video Hits* are essentially collections of recent popular songs translated for computers. *Music Video Hits* includes *Thriller, Let's Hear It for the Boy, Sweet Dreams, Break My Stride, Steppin' Out, Soul Man, Classical Gas,* and *The Hustle. On Stage* features *On the Road Again, Ease on Down the Road, Rise, Duelling Banjos, Cantina Band* (from *Star Wars*), *Entertainment Tonight, Charlie's Angels,* and *Far from Over.*

As the Commodore's SID chip pumps out the music, the "sheet music" appears onscreen. Notes don't scroll by, they sort of flash on and off in musical phrases. After a three-note phrase is finished, the notes disappear and get replaced by the next three. At the bottom of the screen you see the words to each musical phrase, as well as the words to the next phrase.

The music itself has been competently programmed by Peter Engelbrite. Each of the Commodore's three voices can be distinctly heard, which is intriguing to listen to. With most rock music, it's very difficult to sep-

arate the various instruments in your head.

Synthesizers are adding a new dimension to music, as is obvious from recent songs by Herbie Hancock, Thomas Dolby, The Eurhythmics, and others. But simply taking a song and synthesizing it is not necessarily an improvement. Just the opposite, in fact. Digitized *Thriller* is interesting at first, but after a few minutes you get the distinct feeling that you're on an amusement park carousel ride. The music is machinelike, automatic. The human emotion is missing. It's the Muzak-rock of the 1980's.

Whenever we complain about computer-generated *anything,* we can usually add, "But this is more than compensated for by the computer's ability to *interact* with the user." Unfortunately, this isn't the case with Sight & Sound's computer song albums. Hitting the number keys on the top row *will* speed the music up and slow it down. Manipulating the joystick *will* slightly change the "instruments" and create "special effects." But the interaction is minimal. You can't edit the notes to *Soul Man* and add a few licks of your own.

Sight & Sound *does* make two other programs—*The Music Processor* and *The Music Video Kit*—which are compatible and allow more creativity when combined with the computer song albums. Both are sold separately. But if you buy the computer song albums alone, you're basically just getting rehashed rock, at *four* times the cost of a regular album by the original artist.

You've got to *really* like computer music to go for a deal like that. Sight & Sound's selection of songs certainly doesn't make it easy. Admittedly, there are some good tunes here, but each disk is a random hodgepodge of musical styles that is guaranteed to please nobody. Do fans of Joe Jackson's *Steppin' Out* really want to hear *The Hustle*...or vice versa? Willie Nelson fans will certainly get into *On the Road Again,* but do they care about *Ease on Down the Road*? Mason Williams' *Classical Gas* is a good song, but get hip, gang—it's 17 years old! And why anyone would *ever* want to hear computerized renditions of the themes from *Entertainment Tonight* or *Charlie's Angels* defies the imagination. Even *K-Tel* wouldn't release an album with selections like these. It seems as though Sight & Sound just threw on any songs that were available to license.

The heart of rock and roll is still beating, but not on floppies. Real music lovers should stick with the real thing.

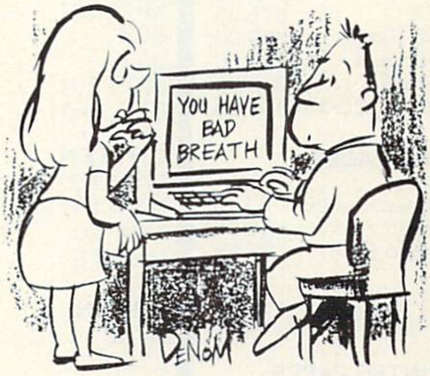
But real music lovers *may* be interested in another disk made by Sight & Sound—*3001 Sound Odyssey.* Many of us are familiar with terms like volume, pitch, waveform, and envelope. These are the physics of sound that make, for example, a trumpet sound different from a piano. Though you may have *heard* of all these terms, chances are you don't fully understand them. Dictionary definitions don't help much when it comes to vague, non-visual concepts.

If only we could manipulate waveform or envelope *ourselves* these concepts would become clear. The computer is the perfect medium to actively experiment with the physics of sound. *3001* isn't a music program like *Music Construction Set* or *Musicalc.* It's a synthesizer that teaches you and shows you how to play with the parameters of sound. The program includes an excellent tutorial that explains all the terms and provides examples of each. After each section, you can get a "Show Me" or a "Try Me" demonstration.

Once you learn the difference between "modulation" and "pulsewidth," switch to the "microsynth" mode and go crazy. A tap of the joystick lets you turn the Commodore's three voices on and off, play with filters and modulation, and manipulate all the elements of sound. Change the waveform from "sawtooth" to "pulse" and see what happens. The top two rows on your keyboard form a mini-piano keyboard that will let you create an unlimited number of sounds, from beautiful music to awful noise. If you're interested at all in the creation of music and sound effects, you'll find *3001 Sound Odyssey* an educational and fascinating experience.

Sight & Sound Music Software, Inc., 3200 South 166th St., New Berlin, WI 53151 (phone: 414-784-5850).

—Dan Gutman



"I think the personal computer is getting a little too *personal."*

WHITE HOUSE COMPUTER

PHONE REBATES! **1-717-322-7700**

P.O. BOX 4025,
WILLIAMSPORT, PA 17701

HOURS: Monday thru Friday 9:00 am till 6:00 pm

VISA and MC
Accepted 4%

DISK DRIVES
C-64
1541 .... 185.95
1571 .... 249.95
1572 .... 375.95
MSD Single 229.00
MSD Dual 459.00
Indus GT 239.00
Cardco SCD-1... 199.00

INTERFACES
Tymac
Connection/2K ... 62.95
Cardco +G 59.95
Cardco B 37.95
OR–D
GPC w/2K 55.95
Grappler CD 89.95

PRINTER PAPER
Lazor 1000 shts... 15.95
Lazor 2500 shts... 29.95

DISKETTES ★
SKC
SS/SD 10.50
SS/DD 13.50
DS/DD 15.50
MAWELL
MD-1 .... 15.95
MD-2 .... 21.95
ELEPHANT
SS/SD 13.50
SS/DD 15.50
DS/DD 19.50

INNOVATIVE CONCEPTS
Flip-N-File
10 .... 3.50
15 .... 8.50
25 .... 18.50
w/lock
50 .... 17.50
50 .... 23.50
ROM Holder... 17.50

KOALA
Pad Disk 39.95
Pad ROM 45.95

★ PRINTERS ★

C-64
MPS 801 .... 179.95
802 .... 208.95
803 .... 175.95
MCS 801 .... 314.95
DPS 1101 .... 314.95

CITIZEN
MSP-10 325
MSP-15 479
MSP-20 469
MSP-25 629

EPSON
RX80 219
LX80 249
FX80 369
JX80 529
RX100 369
FX100 550

C. ITOH
8510 A 285
8510 BC1 329
8510 BC2 389
8510 BP1 339
8510 SP 389
8510 SR 429
8510 SCP 459
8510 SCR 477
1550 BCD 485
1550 P 445
A10-20P 465
F10-40P/S 875
F10-55 P/S 1059
F10 Tractor 169
F10 Sheet Feeder ... 349

MANNSEMAN TALLY
Spirit 80 249
MT 160L 495
MT 180L 595

STAR MICRONICS
SG10 217
SG15 379
SD10 345
SD15 450
SR10 485
SR15 585
Powertype 309

CARDCO
LQ 1 365
★**LQ 3 .... 209**

PANASONIC
1090 187
1091 257
1092 389
1093 585
3151 455

LEGEND
880 209
1080 229
1380 269
1385 305

OKIDATA
182 .... 219.95
84 .... 640.95
92 .... 349.95
93 .... 560.95
Imagewriter 425.95

MONITORS
AMDEK
300G- 119
300A 129
310A 145
Color 300 235
Color 500 345
Color 600 399
Color 700 469
Color 710 539

TEKNIKA
MJ-10 229
MJ-22 285

XTRON
Comcolor I. 189

ZENITH
122 .... 85.95
123 .... 75.95

BMC
Amber 82.95
Green 79.95

COMMODORE
C-1702 .... 199.95
C-1802 .... 204.95
C-1902 RGB 259.95
C-1901 Monchrome ... 129.95



COMPUTERS
C-64 ..... 149.95
C-128 ... 275.95

SOFTWARE ★
BATTERIES INCLUDED
Paper Clip w/Spell Pk ... 69.95
Paper Clip 59.00
Home Pack 32.95
Bus Card II 124.95
B180 Board 99.95

BRODERBUND
Print Shop 29.95
Graphics Library 18.95
B.S. Writer 32.95

EPYX
FAST LOAD 23.95

VIDEO CASSETTE ★★ RECORDER $295.[95]
Symphonic VHS/Remote
- 105 Channel Cable Ready
- 14 Day / 1 Program Timer
- 12 Pre-set Channels
- Auto Rewind

Plus much, much more!

CHECK OUR PRICES!

PRINTER RIBBONS

DUST COVERS

MODEMS
C-1600 .... 44.95
C-1650 .... 54.95
C-1660 .... 54.95
C-1670 ...... 185.95
300/1200 Baud

Cardco
MOD-1 .... 59.95
C-64
MPP-1064 ... 65.95
Tele Learning 49.95

Mitey Mo ... 59.95
Westridge ... 69.95
Compuserve
Starter Kit ... 21.95

WE CARRY A FULL LINE OF:
Batteries Inc., Broderbund, Contineal, Hess, Infocom, Innovative Concepts, Koala, LJK, Microbits, Micropose, Microtek, O.S.S., Professional S.W. Scarborough, S.S.I., Synapse, Time Works, Wico.

POLICY: No deposit on COD orders. FREE freight on all prepaid cash orders over $300 in the continental USA. APO and FPO add $5 per hundred. For priority Mail add $8 per hundred. PA Residents add 6% sales tax. Defective products must have Prior RA number. Schools net 15.

Reader Service No. 173

SCREEN DUMPING ON THE COMMODORE 64, PART II

Etch: A Bit Graphics Sketching Program

Including *Squares* Game for the 64

By Roger S. Macomber
Professor of Chemistry
University of Cincinnati

Probably all of us have, at some time or another, played with Ohio Art's "Etch-a-Sketch." The *Etch* program described here allows you to create similar pictures on your Commodore monitor, but with several additional features. And perhaps most important, it allows you to print the screen image on your printer. (The printing feature was written for a Gemini 10 or 10X printer; use with other printers will require some modification to the program.)

Screen Dumping on the Commodore 64, in the June '85 *Ahoy!*, introduced the principles of screen and printer bit graphics, and demonstrated how to transform a screen bit map into one accessible by an 8-pin dot matrix printer. Readers interested in the details of bit map creation should consult that article. The present program uses similar machine code routines for memory manipulations and printing. But the bulk of the program is written in BASIC and can be readily understood.

To get the program running, here is what you need to do:

1) Move the beginning of your BASIC program with the following instruction: POKE 44, 64 : POKE 16384, 0 : NEW.

2) Enter the program, instructions 1-1190, as it appears on page 97.

3) PEEK in 45 and 46 to determine the location of the end of your program, then POKE in a value 210 greater to accommodate the machine code. (See Part I last month.)

4) RUN the program, and input the 207 bytes of machine code listed at the end of the program. Do this slowly and carefully.*

5) Delete instruction 4.

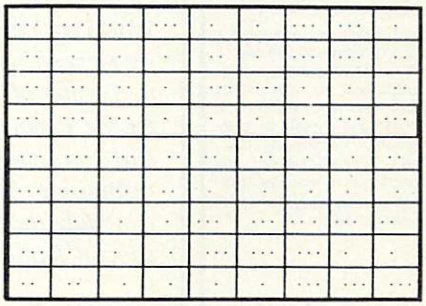
6) SAVE the program, then RUN it.

***If you should ever want to list the machine code at the end of your program, use the following sequence:**

```
4 OPEN 4,4,2 :CMD4 :J=0
5 FOR I = 0 TO 206:PRINT P
EEK(S1+I);:J=J+1
6 IF J>15 THEN PRINT CHR$(
10);:J=0
7 NEXT:PRINT#4,:CLOSE4:STO
P
```

We'll describe the game later, so answer "N" to the first query. You should see a white screen (blue borders), and if you look very carefully at the upper left corner, you'll see one black dot indicating the position of your cursor. You can move this cursor up, down, right, or left using the normal cursor control keys. Notice as you practice moving the cursor that it is prevented from going off the screen. If you press F (Fast) the cursor will move 5 times as fast; pressing S (Slow) will restore the slower movement.

Now, to draw a horizontal or vertical line, press D (Draw) and your cursor will leave a line behind it as it traverses the screen. (When using the Draw function you must continue one dot past the end of the desired line, because the last dot is erased when the draw mode is disabled.) To disable the Draw function, hit D again. Note that if you engage D in the F mode, you will draw a dotted line, while in the S mode, a solid line results. If you make a mistake, you have two options. First, pressing E puts you in the Erase mode and the cursor will erase any dot it crosses. So by retracing the offending line you'll see it disappear. The Erase mode is disabled by pressing E again. Or second, by pressing f7, the entire screen is cleared to begin anew.



Shown: a reduction of an actual dot matrix printout of a game of Squares.

One further feature of the program allows you to draw straight lines at any angle to the horizontal. To draw such a line, bring your cursor to one end of the desired line and press I (Initial), then move your cursor to the other end of the line (anywhere else on the screen except directly above or below the initial point) and press D (Draw) L (Line). Immediately a line is drawn from the final point back to the initial point. Notice also that the D, E and L functions all operate in either the F or S mode.

Finally, when you have created just the picture you want, you can print it (on a Gemini 10 or 10X only!) by simply pressing f1.

SQUARES GAME

Let's return to the game as promised. To do this, it will be necessary to hold down the RUN/STOP key and press RESTORE, then RUN the program again. This time answer "Y" to the prompt. The screen will fill with a 10 dot x 10 dot array, and the cursor dot will appear near the middle of the screen. By using your cursor controls and the Draw function, each player alternates and tries to complete squares in the usual way. When a square is completed, the player identifies it with some number of dots in its middle. The player with the greatest number of squares wins. The printed result of one such game is shown here; the author lost! ☐

SEE PROGRAM LISTING ON PAGE 97

S.O.S.

By Morton Kevelson

Is there any way to suppress the six line perforation skip on the Okimate 10 printer? I have tried changing my printer and interface without any results. I can control the problem with *WordPro 3 Plus/64* by setting the page length to 60 for eleven inch paper. This cannot be done with my *General Ledger Accounts* program from Integrated Software Series (IDI). I have also tried every possible way to access the printer's other text modes from within *WordPro,* but nothing seems to work.

—George Elder
Cornelius, OR

George,

Hold everything. Stop changing hardware with your dealer. There is nothing wrong with your printer or its interface. The Okimate 10 powers up with the perforation skip feature activated. This is described on page 11 of the manual. The manual does not explain how to turn this feature off or on. However, the appropriate command codes are listed in the Command Reference Chart located on the inside back cover. To turn off the automatic perforation skip for use with any of the programs you mentioned, simply execute the following commands prior to running the program:

```
OPEN 4,4
PRINT#4,CHR$(27)CHR$(66)
CLOSE4
```

To turn the perforation skip back on, change the second line to:

```
PRINT#4,CHR$(27)CHR$(65)
```

These commands can be issued in immediate mode as shown, or they can be SAVEd and RUN as a short program by including line numbers.

The difficulty with accessing the Okimate 10 special features from within WordPro *is not a problem with the Okimate 10 printer, but apparently a limitation of* WordPro. *Page 9-6 of the manual indicates that the ability to send special characters in CBM mode is limited to a value of 254. In addition, according to the manual for the Tymac printer interface,* WordPro *disables the special character mode when the "C" option is selected in response to the "Printer?" prompt. As a result, we must conclude that you will not be able to access any of the special features through* WordPro 3 Plus.

I own an MX-80 Epson printer with an 8145 RS232 interface card installed to an HP computer via RS232. I would now like to interface the same printer (without

Continued on page 114

AUTOS: LEASING V. BUYING



For the C-64

By Gil Ragan

Since I had not purchased a new car for several years, the celestial sticker prices came as a complete shock. The high monthly payments and steep interest rates were much more than I had anticipated. The salesman, noting my reaction, urged me to consider leasing. He argued that leasing was actually cheaper than buying. However, I wondered whether his claim was true. Would it be cheaper to lease a new car than buy it?

I struggled with my hand calculator for several long hours before realizing that the problem was far too complex for a few simple calculations. Each auto had a different price and, consequently, a different monthly payment schedule. And there were so many lease plans available. Finally, I turned to my Commodore 64 to find a systematic and thorough way to make the desired comparisons. The result is this *Leasing v. Buying* program. Even if you're not in the market for a new car, you may find this program educational and interesting to run.

The program produces tables showing the cost of owning a new car and the cost of leasing the same car. It tells you what information you need, makes the necessary calculations, and displays the results so you can decide whether it is better to lease or buy in your specific situation. The program can be run over and over so you can easily compare different makes, models, optional equipment configurations, and financing arrangements with their respective lease plans. Although designed for screen display, the program can be modified easily to output to a printer if you wish to study the results.

TYPES OF LEASES

Despite the wide variety of names used, there are basically two types of leases available: closed-end and open- end. Under a closed-end lease, the monthly payments you make are for a specific period of time. When the lease period ends, you return the car and you have no additional financial obligation to the lessor.

Under open-end leases, you agree to pay an additional amount if the value of the car at the end of the lease period is less than the amount estimated at the time the lease is signed. For example, suppose the lessor estimates that the car will be worth $3,850 at the end of the lease period. If its value at that time is only $3,300, you would have to pay the balance of $550. However, if it is worth more than $3,850, you should receive a refund.

This program may be used for both types of leases. However, most leases have a limit on the number of miles you are permitted to drive each year, usually around 15,000. If you exceed this limit, you will have to pay an additional charge for the excess miles. The program assumes that you will stay within the miles limit.

ENTERING THE INFORMATION

After copying the program, type RUN and enter the information requested. Most of the requests are self-explanatory. However, a few words are necessary to insure accurate inputs. Naturally, the buying and lease information should be for the same make and model car in order to produce a meaningful comparison.

The purchase price is the first item requested. For this entry, use the actual price the dealer would be willing to accept, not the manufacturer's suggested retail price. When entering dollar figures, do not use $ or commas. Next you will be asked, WILL YOU NEED FINANCING? Type in a 'Y' if you expect to borrow part of the purchase money. An 'N' will cause the program to branch and skip the questions relating to loans.

For the interest rate request, enter any fractional portion of the rate as a decimal. For example, 15¼ percent should be entered as 15.25. The length of the loan should be entered in months. A four-year loan, for example, should be entered simply as 48 for 48 months. Inputting the length of the loan this way lets the program calculate your monthly loan payments and interest charges. These calculations are made in lines 1030 and 1040.

If you live in a state that imposes a sales tax on autos, enter your state's rate when requested. Don't forget to enter the percentage rate in decimal form. The sales tax in dollars will be computed automatically. For the license fee request, enter your best estimate if you do not know the actual amount.

You are next asked what type of car you are considering. The reason for this request is that cars depreciate in value at

different rates depending upon their types. Average depreciation rates for the three major types of cars are stored in the subroutine in lines 2000-2220. The number you enter here will select the proper group of rates for the type of car you are considering.

One ownership savings frequently overlooked stems from the federal income tax. Interest charges on loans and state sales taxes paid are deductible for tax purposes if you itemize your deductions as opposed to taking the standard deduction (now called zero bracket amount). If you itemize, answer 'Y' to the question posed. This will be followed by a request for your percentage income tax bracket. You can find your income tax bracket by using the X, Y, Z Tables in the income tax booklet which is sent to you with your tax forms each year.

The next section of the program will ask you to enter data about the lease you are considering. First, you will be asked to enter the amount of any *refundable* security deposit you are required to make. This will be followed by a request to enter the amount of the monthly lease payments. Include only the cost of the "pure" lease. If a service maintenance contract is included with your lease package, the cost of this contract should be subtracted to get the actual lease cost.

Some leases require a non-refundable down payment or other charge at the beginning of the lease. If such a payment is required, enter the amount as requested.

THE RESULTS

After all information has been entered, the program will first compute your net cost of buying and will display the results on the monitor. The display format will vary depending upon whether you pay cash or finance the purchase, and upon whether you itemize deductions for income tax purposes or not.

If you do not need to finance your purchase, you will see an item called "Interest lost on purchase price." This item refers to the interest your money could have earned at bank passbook rates had you not bought the car. The interest you give up by spending your funds is one of the hidden costs of ownership. Incidentally, if you have been earning a higher interest rate on your money, insert that rate in line 1015 in place of the .055.

If you finance the purchase, you will see a similar item "Interest lost on down payment" instead. The same reasoning applies to this cost of ownership. The bank passbook rate is in line 1020, and a higher rate can be easily substituted. For financed purchases, an additional line called "Total Payments on Loan" will be displayed. This line shows the total interest charges and loan repayments on the portion of the loan used to purchase the car only. Additional money borrowed for things like the license is excluded to avoid double counting.

Next you will see some deductions. The car you buy should have a value at the end of the loan or lease period. This residual value is an estimate of the amount you should be able to obtain by selling or trading the car at that time. Consequently, it must be deducted to obtain your net cost of buying.

If you itemize your deductions for income tax purposes, the tax savings is computed and also deducted from the total cost of buying to give your net cost.

The program will then compute the cost of leasing. Press 'C' to view the results. Again, note that interest lost on the security deposit and on the non-refundable down payment is computed and added to your cost of leasing. The security deposit will be refunded to you at the end of the lease period, so it is deducted to find your net cost of leasing.

Note that items such as insurance, repairs, maintenance, gas, and oil have been omitted. Under the standard lease, you must pay for these items just as you would as an owner. These items, then, would be the same in both calculations. For this reason, they are ignored, and the program concentrates only on items likely to be different depending on whether you lease or buy.

THE FINAL COMPARISON

Press 'C' once more for a final display. First, you will see what your monthly loan payments will be. This is followed by the monthly lease payment which you entered. Don't be surprised to find that the monthly loan payments are higher than the lease payments. Such a situation will happen particularly if you are making a rather low down payment. This is what the salesmen mean when they imply that leasing is cheaper than buying.

The key comparison is found in the next two lines. They show your net cost of buying and your net cost of leasing. The final line of the program will tell you whether it is cheaper to lease or buy based upon the information you entered.

PRINTER MODIFICATIONS

One final note. If you wish to modify this program to output to a printer, delete lines 520 and 540. These lines merely introduce stops in the screen display. Then, for a VIC-1525 printer, add the following:

```
490 OPEN 2,4
500 CMD 2
750 PRINT#2: CLOSE 2
```

If you have a different printer, consult your manual for the appropriate OPEN, PRINT, and CLOSE statements. □

SEE PROGRAM LISTING ON PAGE 100

WRITING & RUNNING AN ASSEMBLY LANGUAGE PROGRAM

By Mark Andrews

Writing a program in assembly language requires a software package called an assembler-editor, or simply an assembler. Three of the most popular assemblers for the Commodore 64 are:

• The *Commodore 64 Macro Assembler Development System,* manufactured by Commodore.

• The *Merlin 64* assembler, manufactured by Roger Wagner Publishing, Inc. (formerly Southwestern Data Systems) of Santee, CA.

• The *Panther C-64* assembler, manufactured by Panther Computer Corporation of Los Angeles.

It wouldn't be fair to rate these in 1-2-3 order, since each has its advantages and disadvantages. But to help you decide which you'd like to use, here are brief descriptions of each:

THE COMMODORE 64 ASSEMBLER

An excellent package, designed by professionals for professionals, the *Commodore 64 Macro Assembler Development System* (the *Commodore 64 Assembler* for short) works much like the large assemblers used with mainframes. Unfortunately, though, it isn't what I'd call user-friendly; in fact, it can be downright user-hostile, especially for people who are just starting to study assembly language.

The *Commodore 64 Assembler* comes on a single 5¼-inch disk. But there are several programs on that disk, and each time you write and assemble a program, you have to load and run several individual utilities—in the correct order. If you use the programs in the wrong sequence, or make a mistake in using one program before you load the next one, you sometimes have to start the process all over again. That can be both annoying and time-consuming.

THE MERLIN 64

The *Merlin 64* is an imaginatively designed assembler with a host of advanced features—some not available in any other C-64-compatible assembler system. For example, *Merlin* comes with a very sophisticated disassembler (a utility that can convert machine language into assembly language). Another bonus is a large library of useful assembly language routines that can be incorporated into user-written programs.

Merlin is equipped to handle macros—prewritten routines that can be easily inserted into assembly language programs—and has linking capabilities that enable the user to write programs ordinarily too long to fit into a Commodore 64's memory. And *Merlin* can be used in an 80-column format on a computer equipped with a high-resolution monitor and an 80-column card.

THE PANTHER C-64 ASSEMBLER

The *Panther C-64 Assembler* is even easier to use than *Merlin*—so it's a popular package, especially among newer assembly language programmers. But it lacks some features that have become almost standard in assemblers—macro capabilities, for example—and for some reason, the displays and the programs listings it creates are always printed in lower-case letters, a feature I find annoying. Furthermore, the *Panther* consumes more memory space than either the *Commodore 64 Assembler* or *Merlin 64*, and some of the memory space the assembler eats up is in blocks of RAM which I feel should be left free for user-written programs.

On the positive side, the *Panther* is a joy to use; everything that it can do is directly accessible from the editor (a costly feature in terms of memory), and its assembly language editor has a beautifully engineered error-checking system that makes it almost impossible to write a bad line of code. And the instruction manual that comes with the program is very good; in fact, if I didn't own a *Panther* assembler, I might be tempted to buy one just to get my hands on the tutorials in its manual.

GOLDILOCKS & THE 3 ASSEMBLERS

When I started writing this series of columns, it was difficult to decide whether to use the *Commodore 64 Assembler, Merlin* or the *Panther C-64.* So I picked one by the Goldilocks method. After much experimentation, I concluded that the *Commodore 64 Assembler* was too big, the *Panther* was too little, and the *Merlin 64* was just about right. So most of the programs you'll see in this column were created with *Merlin.* If you own some other kind of assembler, that's okay; with minor modifications, the programs in these columns will work with any assembler compatible with the Commodore 64. But

for consistency, they'll all be written on a *Merlin*—and the column you're reading now, a tutorial on assemblers, will focus on the *Merlin 64*.

And that brings us to our real topic: how to use an assembler (such as the *Merlin*) to write a Commodore 64 assembly language program.

USING THE MERLIN 64

Merlin 64 can be loaded and executed like any other disk-based program. When you've successfully booted *Merlin*'s master disk, this menu will be displayed on your monitor screen:

```
C : Catalog
L : Load source
S : Save source
R : Read text file
W : Write text file
D : Drive change
E : Enter ED/ASM
O : Save object code
G : Run program
X : Disk command
Q : Quit
```

The choices on this menu can be used to instruct *Merlin* to do quite a few things—from loading and saving assembly language programs (choices "L" and "S") to listing the names of the files a disk ("C") to running a machine language program ("G"). You can even format disks, scratch files from disks, and perform numerous other disk-management functions by choosing "X".

To write an assembly language program, you have to pick menu choice "E". That will that *Merlin* out of its "executive" (menu) mode and put it into its editor-assembler mode, which can be used for both writing and assembling assembly language programs.

When you've made menu choice "E" and put *Merlin* into editor-assembler mode, the menu on your screen will disappear, and you'll see a ":" prompt at the top of your screen. After that prompt, type

```
A
```

—for "append." *Merlin* will display the number "1" on your screen. That "1" is an automatically generated line number, and as soon as it appears, you can start typing an assembly language program.

Let's pause for a word about line numbers. One *Merlin* feature that takes some getting used to is its automatically generating line numbers, beginning with number 1 and progressing in increments of 1. So you never have to type a line number when you're using the program. And when you add or delete lines, all affected line numbers will change automatically.

When *Merlin* generates and displays a line number, the number is always followed by a space and a flashing cursor. If you have *Merlin* up and running now, and if that's what you see on your screen, type an asterisk—*without* any additional spaces in front of it. Line 1 of your program should then look like this:

```
1 *
```

Merlin will then advance automatically to Line No. 2. Following the numeral 2, again without any extra spacing, type

```
* ADDNRS
```

—and hit your RETURN key. Then, when *Merlin* advances to Line 3, type another asterisk.

This is what you should see on your screen now:

```
1 *
2 * ADDNRS
3 *
4
```

In a moment, we'll discuss what those lines mean. First, though, let's get a little more familiar with *Merlin*.

At Line 4, press your carriage return, and you'll see *Merlin*'s ":" prompt again. Then you can type "A" (for ADD) again, and continue writing your program. Or if you prefer, you can type some other command: "L" for list, for example. This will list your program, in its entirety, on your screen.

Another command that can follow the ":" prompt is "D" for DELETE. You would type the letter "D" followed by the number of the line (or lines) you want to delete. Suppose you wanted to delete Lines 2 and 3 in the above listing. You would simply type

```
D2,3
```

after the ":" prompt. Try it! Then restore the lines you've deleted by using the "A" command.

Still another command that can be used after the ":" prompt is "I" (for INSERT). Type the letter "I" followed by the number of the line where you want your new line inserted. Suppose you wanted to insert another asterisk at Line 2 in the above program. You could simply hit your RETURN key to get a colon prompt, then type

```
I2
```

Try that, and you'll see *Merlin* respond with the number

```
2
```

Now type an asterisk, followed by two carriage returns. *Merlin* will display its ":" prompt again, and you can then type "L" for list. Then *Merlin* will list your program, and you'll see that another line containing an asterisk has indeed been inserted into your program, at Line 2.

Speaking of lines, you can now delete that extra asterisk you've just added to your program. Hit a carriage return to get a colon prompt and type "D2." Then you can type "L" for LIST, and you'll get a listing showing you that your program looks like this again:

```
1 *
2 * ADDNRS
3 *
4
```

That looks okay, so now you can continue typing until you've entered the following program into your computer:

```
1 *
2 * ADDNRS
3 *
4          ORG   $8000
5 ADDNRS   CLD
6          CLC
7          LDA   #2
8          ADC   #2
9          STA   $02A7
10         RTS
11
```

LISTING YOUR PROGRAM

When you've reached Line 12 in your ADDNRS program, hit your RETURN key. Then you can type either L or LIST, and the complete program will be listed on your screen. Then you can examine the program line by line.

Lines 1 through 3 of the ADDNRS program are *comments.* Line 2 explains what the program does, and lines 1 and 3 set off the explanatory line by printing asterisks followed by white space.

Line 4 is the origin line of the ADDNRS program. Every program written on *Merlin 64* must start with an origin line. When a computer runs a machine language program, the first thing it does is go to a predetermined memory location and look at the value stored at that address. So when you write an assembly language program, the first thing *you* have to do is tell your computer where to start looking for the program in its memory.

The origin directive looks like a simple line to write; but deciding where to start a program can be a difficult task. There are many blocks of memory in your computer that you can't use for assembly language programs because they're reserved for other uses–for example, to hold your computer's operating system, disk operating system and BASIC interpreter. The best way to learn your way around your Commodore's memory banks is to consult a lot of memory maps and write a lot of assembly language programs. But in the early stages, it will usually be safe to start your programs somewhere around memory location $8000 (or 32768 in decimal notation).

Line 5 of the ADDNRS program contains the label ADDNRS and the assembly language instruction CLD.

Labels always occupy the *first field* (the column that follows the line-number column) in an assembly language programs. And labels are very important in Commodore 64 assembly language programs, since labels, not line numbers, are used to access routines.

The abbreviation CLD in Line 5 is a *mnemonic,* or assembly language instruction. Mnemonics always occupy the second field (after the line number) in assembly language programs. The mnemonic CLD, which means "clear decimal mode," is often used prior to arithmetical operations in Commodore 64 assembly language. Inside your computer's 6510 microprocessor, a status flag called the decimal flag can be cleared with the instruction CLD. When the decimal flag is cleared, the Commodore 64's 6510 chip carries out all arithmetical operations using binary numbers. If the decimal flag is not cleared prior to an arithmetical operation, the operation is carried out using binary-coded decimal (BCD) numbers, a subject that will be discussed in later columns.

In *Line 6* of the ADDNRS program, the mnemonic CLC is used to clear another 6510 flag, called the carry flag. This flag, as explained last month, is affected by so many kinds of operations that it's good programming practice to clear it before every addition operation–and to *set* it before every subtraction operation, so that it can be used as a borrow.

Line 7 of the ADDNRS program–"LDA #2"–is a very

straightforward instruction. When two numbers are to be added in an assembly language program, one must be loaded into a special 6510 register called the accumulator. Then the second must be added to the number that has just been placed in the accumulator. The sum of the two numbers is then left in the accumulator.

In *Line 7* of the ADDNRS program, the instruction LDA #2 is used to load the literal number 2 in the accumulator. The "#" sign in front of the number 2 means that it's a literal number, not an address. If the instruction in Line 7 were "LDA 2," the accumulator would be loaded with the contents of Memory Address 0002, not the number 2.

Line 8–"ADC #2"–adds the literal number 2 to the number in the accumulator (in this case, another 2). As mentioned, there is no 6510 assembly language instruction that means "add without carry." So the only way an addition operation can be performed without a carry is to clear the status register's carry flag and perform an "add with carry" operation.

Line 9–"STA $02A7"–completes the addition operation in the ADDNRS program. This statement stores the contents of the accumulator in Memory Address $02A7 (usually a safe spot for storing data). Note that the symbol "#" is not used before the operand ($02A7), since in this case the operand is a memory address, not a literal number.

The mnemonic RTS, in *Line 10* of the ADDNRS program, means "Return from Subroutine." The instruction RTS has two main uses in assembly language. When used at the end of a subroutine, it works like the RETURN instruction in BASIC; it ends the subroutine and returns to the main body of a program, beginning at the line following the line in which the RTS instruction appears. But if used at the end of the main body of a program–as it is here–it ends the program. Usually, control of the computer then returns to whatever was in control before the program began–usually the Commodore 64's built-in BASIC interpreter.

ASSEMBLING AND SAVING YOUR PROGRAM

To assemble the ADDNRS program using *Merlin*, all that is necessary is to type the command ASM following the ":" prompt. *Merlin* will then ask you if you want to update your source-code file–with the current date, for example. If you don't want to do that, you can type "N" (for "no") and *Merlin* will assemble your source-code program–*very* rapidly!

After youu've assembled a source-code listing into object code, you can save both your source-code and object-code listings to disk. Here's how:

First type "Q" (for QUIT) after the ":" prompt to get your assembler back into its "executive" (menu) mode. *Merlin*'s main menu will then reappear. You can then save your source code by selecting menu choice "S," and your object code by picking menu choice "O."

After you type "S" or "O," *Merlin* will ask you what you'd like to name your program. You don't have to add a suffix to indicate whether it's a source-code listing or an object code program; *Merlin* will automatically add an "S" to the name of a source code listing, or "O" to the name of an object code listing.

RUNNING AN ASSEMBLY LANGUAGE PROGRAM

Once an assembly language program has been assembled into machine language, it can be executed without an assembler. When you have ADDNRS program assembled and safely stored on a disk, you can turn your computer off, turn it on again, and type the command

```
LOAD "ADDNRS.O",8,1
```

(As you may know, the number "1" at the end of a "LOAD" command informs your computer that the program being loaded is in machine language, and must therefore be loaded starting at the address that was designated in the program's ORG line.)

When the ADDNRS.O program has been loaded into memory, you can run it by typing the command

```
SYS 32768
```

That line will instruct your computer to run the program that now begins at Memory Address 32768 ($8000 in hexadecimal notation). And, since the ADDNRS.O program has just been loaded into memory starting at that address, the program that will now be executed is the ADDNRS program.

AND NOW FOR SOMETHING COMPLETELY DIFFERENT

Unfortunately, the ADDNRS program doesn't appear to do much when it is loaded into memory and executed. All it does is add a couple of numbers and place their sum into a certain memory location; and that isn't very exciting, since no keyboard inputs are requested, and nothing is displayed on your computer screen.

So, in the listing on page 96, I've provided something to tide you over until next month's column. It's considerably more interesting than the ADDNRS program, and it will give you some practice in typing, assembling, saving and running assembly language programs. Type the *Flash* program on page 96, assemble it, and save it on a disk. Then run it, and watch the action. We'll discuss how it works in next month's column. □

SEE PROGRAM LISTING ON PAGE 96

Next month: ***Addressing the Commodore, Part I*–the first of two columns that will explain the 13 addressing modes used in Commodore 64 assembly language programming, including several short illustrative programs.**

COMMODARES

PROGRAMMING CHALLENGES

By Dale Rupert

Each month, we'll present several challenges designed to stimulate your synapses and toggle the bits in your cerebral random access memory. We invite you to send your solutions to:

Commodares c/o *Ahoy!*
P.O. Box 723
Bethel, CT 06801

We will print and discuss the cleverest, simplest, shortest, most interesting and/or most unusual solutions. Be sure to identify the *name* and *number* of the problems you are solving. Also show sample runs if possible, where appropriate. Programs on diskette are welcome, but they must be accompanied by listings. Also tell what makes your solutions unique or interesting, if they are. You must enclose a stamped, self-addressed envelope if you want any of your materials returned.

Your original programming problems, suggestions and ideas are equally welcome! The best ones will become *Commodares.*

Problem #19-1: Binary Palindrome

Write a program to determine whether the binary equivalent of any input decimal number between 0 and 255 is a palindrome or not. Just one catch. Your program must *not* calculate the binary equivalent of the given number. (The eight-bit binary number must read the same forward or backward to be a palindrome.)

Problem #19-2: Memory Locator

This program does nothing more than tell where it is located in memory with an output such as this:

Program Text : AAA – BBB
Numeric Variables : CCC – DDD
Arrays : EEE – FFF
String Storage : GGG – HHH

where for example AAA and BBB are the starting and ending memory addresses of the program's text.

This is an easy problem if you know how to do it. Then again, aren't they all?

Problem #19-3: Orthogonal Time

At 3 o'clock and 9 o'clock the hands of the clock form a ninety degree angle. At what other times do the hands form a right angle?

If you are tempted to say 12:15, consider the fact that the hour hand has moved by the time the minute hand gets to the three. Your program should list all the times to the nearest second at which the hands of a clock are ninety degrees apart. If you submit your program, also send the answers it gives.

Problem #19-4: Screen Scramble

Jesus Geliga-Torres (Aguadilla, PR) suggested the following *Commodare.* Write a program which takes whatever is on the screen and rewrites it in a top to bottom, right to left fashion. The top row of the screen becomes the right hand column. Whatever was in the upper left corner is now in the upper right corner. The contents of the lower left corner will now be in the upper left corner. This should work for any text or text-mode graphics on the screen. !nuf ekil sdnuoS

First we will discuss a solution to last month's *Problem #18-1: Cycling Function* from Jim Speers (Niles, MI), then we will look at other readers' solutions to *Commodares* from the March issue of *Ahoy!*

Jim's solution to the problem he submitted is as follows:

```
10 DEF FNC(D)=D - 2*5*(X=0) + 2*5*(X=100
)
```

When this statement is combined with the statements given last month

```
20 D=5 : X=0
30 PRINT X : X=X+D : D=FNC(D) : GOTO 30
```

the output cycles back and forth from 0 to 100 in steps of five. Line 10 is written so that you may replace the fives with any other step size you desire. Line 20 should be modified accordingly. Also you must be sure that the test values in line 10 (0 and 100) are reached exactly. If not, the output doesn't cycle. You might change the "=" to "<=" and ">=" if you are sure whether the endpoints will be reached exactly.

Here are two more general cycling functions which Jim suggested:

```
10 FNU(V) = V - S*(V<MX) + (MX-MN)*(V=MX
)
```

```
15 FND(W) = W + S*(W>MN) - (MX-MN)*(W=MN
)
```

MN, MX, and S are the minimum value, maximum value, and step size. FNU repeatedly cycles up from the minimum to the maximum value and then drops back to the minimum value to start over. FND does just the opposite. Add line 5 and change lines 20 and 30 as follows to see how these functions work:

```
5 MN=45 : MX=99 : S=3

20 X=MN : Y=MX

30 PRINT X,Y : X=FNU(X) : Y=FND(Y) : GO
TO 30
```

Again, the endpoints must be reached exactly with the chosen step size.

Problem #15-1: Fancy Functions brought many responses. Readers used two significantly different methods to check for the "evenness" of a number as typified by these function definitions:

```
FNA(X) = -(X/2 = INT(X/2))
```

```
FNA(X) = 1 - (X AND 1)
```

The first uses a logical expression having a value −1 when true and 0 when false. The minus sign in front produces the desired result that FNA(X) equals 1 when X is even and 0 when X is odd. The second uses a Boolean expression which depends on the fact that the least significant bit in the binary representation of an even number is 0 and of an odd number is 1. (X AND 1) equals 1 for odd values of X and 0 for even values of X. Refer to the *Commodore 64 Programmer's Reference Guide* for a discussion of the AND operator.

A solution to the equilateral triangle area function is

```
FNB(X) = X*X*SQR(3)/4
```

or some variation on this theme.

There was not much need for originality for the random number generator function. Most solutions looked essentially like this:

```
FNC(X) = INT(X*RND(1)+1)
```

where FNC has a random integer value from 1 to X inclusive.

The fourth function as defined by most readers was of the form

```
FND(X) = INT(N*10^X + .5)/10^X
```

This rounds the number N properly to X decimal places, but only if X is positive. A more general solution handles negative numbers as well:

```
FND(X) = SGN(X)*INT(ABS(N)*10^X+.5)/10^X
```

Glenn D. Elliot (Rutherford, NJ) used string functions to arrive at his solution:

```
FND(X) = VAL(LEFT$(STR$(N+5/10^(X+1)),LE
N(STR$(INT(N)))+X+1))
```

Unfortunately, this doesn't work properly for values of N less than 1. Can someone fix it up? How about a function that returns a value with X significant figures? For example, if N=1.254 and X=2, the result is 1.3, or if N=1254 and X=2, the result is 1300 or 1.3E3. Anyone up for the challenge? The string approach might be the way to go.

Several readers overlooked the requirement for *Problem #15-2: Prime Factors* that the solution contain only one statement per line. It is surprising how the inability to put more than one statement per line can complicate a program, especially with IF-THEN statements. The solution from John Prager (Bay City, MI) printed below is representative of most of the other proper solutions.

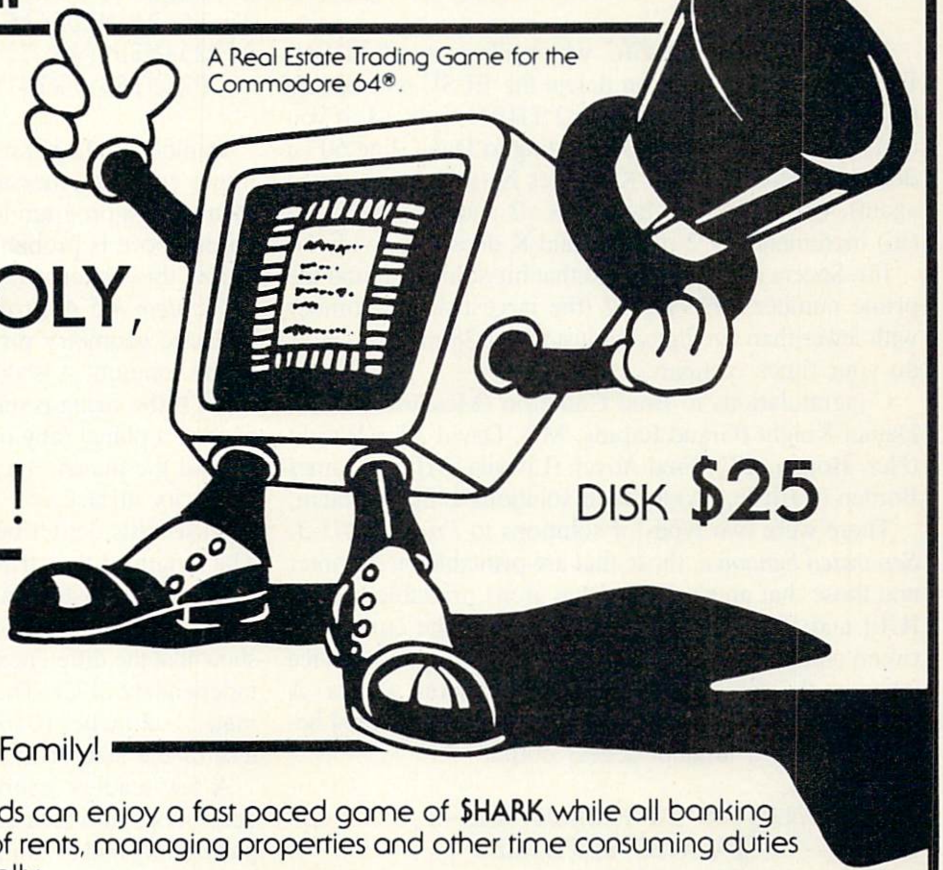
```
10 REM SOLUTION TO PROBLEM #15-2 :
20 REM       PRIME FACTORS
30 REM    BY  JOHN PRAGER
100 INPUT N
110 FOR F=2 TO SQR(N)+.9
120 IF INT(N/F)=N/F THEN 150
130 NEXT F
140 F=N
150 IF A THEN PRINT"*";
160 PRINT F;
170 A=-1
180 N=N/F
190 IF N>1 THEN 110
```

John used the fact that you don't have to check for any factors larger than the square root of the given number. Any non-prime non-square number can be factored into factors, one of which is larger and the other of which is smaller than the square root of the number. Once you've found the smaller factor, division gives the larger. One way to speed this program up is to check only for odd factors once all factors of two have been divided up. The solution from Dana Rousseau (Memphis, TN) below takes that approach.

```
1 REM SOLUTION TO PROBLEM #15-2 :
2 REM        PRIME FACTORS
3 REM    BY DANA ROUSSEAU
4 REM   (( SIMON'S BASIC ))
10 INPUT N
20 K=2
30 L=N-K*INT(N/K)
40 IF L=0 THEN PRINT K;"*";
```

$HARK™

A Real Estate Trading Game for the Commodore 64®.

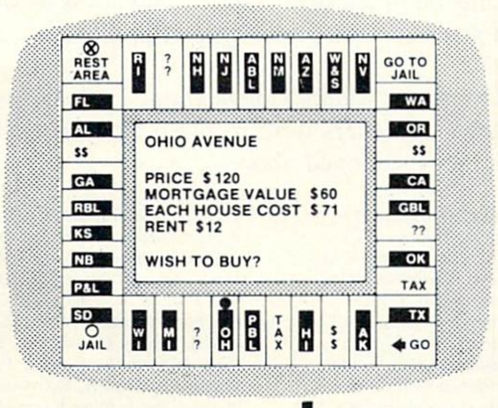


If you enjoy playing MONOPOLY, you'll love $HARK!

DISK $25

It's Fun For the Whole Family!

Now you and your friends can enjoy a fast paced game of **$HARK** while all banking transactions, collecting of rents, managing properties and other time consuming duties are handled automatically.



With **$HARK**, you can:

- Buy and mortgage property
- Build and tear down houses
- List your cash, property and net worth
- Make **DEALS** and **TRADES** with other players
- Include or void different game options
- Have up to 6 people play at one time

(Screen Simulation)

SAGE SOFTWARE
P.O. Box 2578-F
Freeport, Tx 77541-2578

Our Guarantee: If you are not satisfied with your disk, please return to Sage Software for a free replacement.

$HARK™

A Real Estate Trading Game

YES, please send me **$HARK**

___ Disk(s) @ $25 (U.S.) each

Send check or money order to:
SAGE SOFTWARE
P.O. Box 2578-F
Freeport, TX 77541-2578

(Please Print)

Name ________________ Address ________________

City ________________ State ________________ Zip ________

Sage Software promises prompt delivery!

```
50 IF L=0 THEN N=N/K
60 IF K*L>2 THEN K=K+2: ELSE :K=K+L
70 IF K*K<=N THEN 30
80 PRINT N
```

Dana used *Simons' BASIC*, which allows the IF-THEN-ELSE statement. You can delete the ELSE part of line 60 and add line 65 IF K*L<=2 THEN K=K+L if you don't use *Simons' BASIC*. According to Dana, line 60 (i) does not increment K if K divides N (so it can be tried again), (ii) increments by 1 if K=2 and N is odd, and (iii) increments by 2 if K>2 and K does not divide N.

Jim Speers (Niles, MI) said that his solution found the prime number 999,999,937 (the largest prime number with fewer than ten digits) in just under 38 minutes. How do your times compare?

Congratulations to Brad Cameron (Metcalfe, ONT), Daniel Knight (Grand Rapids, MI), David Alan Wright (New Britain, CT), Fred Atiyeh (Livonia, MI) and James Borden (Carlisle, PA) for their solutions to this problem.

There were two types of solutions to *Problem #15-3: Separated Sentence*, those that are printable on a printer and those that aren't. Those that aren't printable use the [UP] and [DOWN] statements to move the cursor between characters. A printer is a sequential access device whereas the screen (or a plotter) is random access. A one-liner from Tom Nielsen (Bismarck, ND) is listed below. It is for a random access output.

```
1 REM SOLUTION TO PROBLEM #15-3 :
2 REM    SEPARATED SENTENCE
3 REM   BY TOM NIELSEN
4 REM
10 INPUTB$:FORI=1TOLEN(B$)STEP2:PRINTMID
$(B$,I,1)""MID$(B$,I+1,1)"";:NEXT:PRINT"
"
```

Of the solutions which are hard-copy printable, some stepped through the input string twice, printing every other letter on one line, then dropping to the next line to print the rest. An example of this method is the program listed here from Marcus F. Cooper (Mililani, HI).

```
1 REM SOLUTION TO PROBLEM #15-3 :
2 REM    SEPARATED SENTENCE
3 REM   BY MARCUS F. COOPER, JR.
4 REM
10 INPUTS$:FORU=1TOLEN(S$)STEP2:PRINTMID
$(S$,U,1)+" ";:NEXT
20 PRINT:FORL=2TOLEN(S$)STEP2:PRINT" "+M
ID$(S$,L,1);:NEXT
```

Other readers created separate strings and printed them individually, as shown by this solution from Larry E. Cox (Tecumseh, MI).

```
1 REM SOLUTION TO PROBLEM #15-3 :
2 REM    SEPARATED SENTENCE
3 REM   BY LARRY E. COX
4 REM
10 INPUT"SENTENCE";A$
20 FORX=1TOLEN(A$)STEP2:Y=X+1
30 B$=B$+MID$(A$,X,1)+CHR$(32):C$=C$+CHR
$(32)+MID$(A$,Y,1):NEXT
40 PRINTB$:PRINTC$
```

Daniel Knight (Grand Rapids, MI) suggested that you might encode a message with dummy letters this way, then write a program to decode it. The third type of solution above is probably the most useful if you want to utilize this concept in cryptography.

Problem #15-4: String Stretcher was meant to revive a classic geometry problem. Even after seeing a proof of the solution, it is still a bit difficult to believe the results. If the string is one yard longer than the circumference of a planet (any planet), the string could be placed around the planet at a uniform height of 1/(2*PI) yards above its surface.

To see this, let C1 be the circumference of the planet. The length of the string is then C1 plus one yard. The radius of the planet is C1/(2*PI) and the radius of the string is (C1+1)/(2*PI). Some algebraic manipulations show that the difference in the two radii is simply 1/(2*PI), independent of C1. Therefore the string will be approximately six inches (0.16 yards) above the surface regardless of the size of the planet.

A few readers resorted to some craftiness by putting delay loops into their solutions so it looked like the computer really had to "think" about the solution before printing it. Of the readers who commented, all said they enjoyed this bit of trickery. Your comments as to whether or not you like a particular type of problem are always welcome.

Solutions for March *Commodares* and various other problems and letters were received from the following people not mentioned above:

Scott Kruger (Rockford, IL)
Tony Ruperto (Kitimat, BC)
Stan Beddingfield (Dover, DE)
Rick Acosta (Somerville, NJ)
Don Allen (Necedah, WI)
Jason Simpson (Victoria, BC)
Russ Beinder (Victoria, BC)
Keith Rasmussen (Blaine, MN)
Paul Hawthorne (Wilburton, OK)
Peter Lam (Langenburg, SK)
R. Dorn (Saratoga, NY)
Tom McNeill (Baltimore, MD)
Frank Gourley (Kenesaw, NE)
Brad Cameron (Metcalfe, ONT)
Kenneth Karow (Chicago, IL)
Steve Feld (New York, NY)
Veronica Robin (Niles, OH)
James N. Martin (Keesler AFB, MS)
Richard Cannon (Cambridge, ONT)
William Ingram (Cambridge, ONT)
Trevor Green (Campsie NSW, Australia)
Glen Oldford (St. John's, NFLD)
Eric Berns (Grande Prairie, ALB)
Mike Livermore (Kansas City, KS)
James Dunavant (Gainesville, FL)
Vadim Israilevich (N. Hollywood, CA)
Margaret Wylde (New Bloomfield, MO)
Walter Robinson (Bonner Springs, KS)
John Northover (San Diego, CA)
Cathy Levandofsky (Bainbridge, OH)
Carl Lancaster (Oxon Hill, MD)
L.M. Davis (Bainbridge, OH)
Stanley Davis (Decatur, GA)
Paul Lalli (McAlester, OK)

As usual, solutions received by the middle of the month of the magazine's cover date are most likely to be listed and discussed, but you may send your solutions any time. I will respond to your questions or comments only if you enclose a self-addressed stamped envelope. You have some serious work ahead of you with this month's challenges, so you'd better get busy! □

PROGRAM LISTINGS

Attention new *Ahoy!* readers! You must read the following information very carefully prior to typing in programs listed in *Ahoy!* Certain Commodore characters, commands, and strings of characters and commands will appear in a special format. Follow the instructions and listing guide on this page.

On the following pages you'll find several programs that you can enter on your Commodore computer. But before doing so, read this entire page carefully.

To insure clear reproductions, *Ahoy!*'s program listings are generated on a daisy wheel printer, incapable of printing the commands and graphic characters used in Commodore programs. These are therefore represented by various codes enclosed in brackets [ ]. For example: the SHIFT CLR/HOME command is represented onscreen by a heart . The code we use in our listings is [CLEAR]. The chart below lists all such codes which you'll encounter in our listings, except for one other special case.

The other special case is the COMMODORE and SHIFT characters. On the front of most keys are two symbols. The symbol on the left is obtained by pressing that key while holding down the COMMODORE key; the symbol on the right, by pressing that key while holding down the SHIFT key. COMMODORE and SHIFT characters are represented in our listings by a lower-case "s" or "c" followed by the symbol of the key you must hit. COMMODORE J, for example, is represented by [c J], and SHIFT J by [s J].

Additionally, any character that occurs more than two times in a row will be displayed by a coded listing. For example, [3 "[LEFT]"] would be 3 CuRSoR left commands in a row, [5 "[s EP]"] would be 5 SHIFTed English Pounds, and so on. Multiple blank spaces will be noted in similar fashion: e.g., 22 spaces as [22 " "].

Sometimes you'll find a program line that's too long for the computer to accept (C-64 lines are a maximum of 80 characters, or 2 screen lines long; VIC 20 lines, a maximum of 88 characters, or 4 screen lines). To enter these lines, refer to the *BASIC Command Abbreviations Appendix* in your User Manual.

On the next page you'll find our *Bug Repellent* programs for the VIC 20 and C-64. The version appropriate for your machine will help you proofread our programs after you type them. (Please note: the *Bug Repellent* line codes that follow each program line, in the whited-out area, should *not* be typed in. See the instructions preceding each program.)

Also on the following page you will find *Flankspeed*, our ML entry program, and instructions on its use. ☐

Call *Ahoy!* at 212-239-0855 with any problems.

When You See	It Means	You Type	You Will See
[CLEAR]	Screen Clear	SHIFT CLR/HOME	
[HOME]	Home	CLR/HOME	
[UP]	Cursor Up	SHIFT ↑CRSR↓	
[DOWN]	Cursor Down	↑CRSR↓	
[LEFT]	Cursor Left	SHIFT ←CRSR→	
[RIGHT]	Cursor Right	←CRSR→	
[SS]	Shifted Space	SHIFT Space	
[INSERT]	Insert	SHIFT INST/DEL	
[DEL]	Delete	INST/DEL	
[RVSON]	Reverse On	CNTRL 9	
[RVSOFF]	Reverse Off	CNTRL 0	
[UPARROW]	Up Arrow	↑	
[BACKARROW]	Back Arrow	←	
[PI]	PI	π	
[EP]	English Pound	£	

When You See	It Means	You Type	You Will See
[BLACK]	Black	CNTRL 1	
[WHITE]	White	CNTRL 2	
[RED]	Red	CNTRL 3	
[CYAN]	Cyan	CNTRL 4	
[PURPLE]	Purple	CNTRL 5	
[GREEN]	Green	CNTRL 6	
[BLUE]	Blue	CNTRL 7	
[YELLOW]	Yellow	CNTRL 8	
[F1]	Function 1	F1	
[F2]	Function 2	SHIFT F1	
[F3]	Function 3	F3	
[F4]	Function 4	SHIFT F3	
[F5]	Function 5	F5	
[F6]	Function 6	SHIFT F5	
[F7]	Function 7	F7	
[F8]	Function 8	SHIFT F7	

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** This and the preceding explain these codes and provide other essential information on entering **Ahoy!** programs. Read these pages **before** entering programs.

BUG REPELLENT

This program will let you debug any *Ahoy!* program. Follow instructions for VIC 20 (cassette or disk) or C-64.

VIC 20 VERSION

By Michael Kleinert and David Barron

For cassette: type in and save the *Bug Repellent* program, then type RUN 63000[RETURN]SYS 828[RETURN]. If you typed the program properly, it will generate a set of two-letter line codes that will match those listed to the right of the respective program lines.

Once you've got a working *Bug Repellent*, type in the program you wish to check. Save it and type the RUN and SYS commands listed above once again, then compare the line codes generated to those listed in the magazine. If you spot a discrepancy, a typing error exists in that line. Important: you must use exactly the same spacing as the program in the magazine. Due to memory limitations on the VIC, the VIC *Bug Repellent* will register an error if your spacing varies from what's printed.

You may type SYS 828 as many times as you wish, but if you use the cassette for anything, type RUN 63000 to restore the *Repellent*.

When your program has been disinfected you may delete all lines from 63000 on. (Be sure the program you type doesn't include lines above 63000!)

For disk: enter *Bug Repellent*, save it, and type RUN:NEW [RETURN]. Type in the program you wish to check, then SYS 828.

To pause the line codes listing, press SHIFT.

To send the list to the printer type OPEN 4,4:CMD 4:SYS 828[RETURN]. When the cursor comes back, type PRINT#4:CLOSE 4[RETURN].

```
•63000 FORX=828TO1023:READY:POKEX,Y:NEXT:END   AC
•63001 DATA169,0,133,63,133,64,165,43,133,251  JL
•63002 DATA165,44,133,252,160,0,132,254,32,228 DF
•63003 DATA3,234,177,251,208,3,76,208,3,230    OE
•63004 DATA251,208,2,230,252,169,244,160,3,32  OH
•63005 DATA30,203,160,0,177,251,170,230,251,20
 8                                             KO
•63006 DATA2,230,252,177,251,32,205,221,169,58 JJ
•63007 DATA32,210,255,169,0,133,253,230,254,32 OK
•63008 DATA228,3,234,165,253,160,0,170,177,251 LG
•63009 DATA201,32,240,6,138,113,251,69,254,170 BP
•63010 DATA138,133,253,177,251,208,226,165,253
 ,41                                           DD
•63011 DATA240,74,74,74,74,24,105,65,32,210    EK
•63012 DATA255,165,253,41,15,24,105,65,32,210  FO
•63013 DATA255,169,13,32,210,255,173,141,2,41  PK
•63014 DATA1,208,249,230,63,208,2,230,64,230   CB
•63015 DATA251,208,2,230,252,76,74,3,169,236   KH
•63016 DATA160,3,32,30,203,166,63,165,64,32    DP
•63017 DATA205,221,169,13,32,210,255,96,230,25
 1                                             EL
•63018 DATA208,2,230,252,96,0,76,73,78,69      OI
•63019 DATA83,58,32,0,76,73,78,69,32,35        FG
•63020 DATA32,0,0,0,0,0                        LE
```

C-64 VERSION

By Michael Kleinert and David Barron

Type in, SAVE, and RUN the *Bug Repellent*. Type NEW, then type in or LOAD the *Ahoy!* program you wish to check. When that's done, SAVE your program (don't RUN it!) and type SYS 49152 [RETURN].

To pause the listing depress and hold the SHIFT key.

Compare the codes your machine generates to the codes listed to the right of the respective program lines. If you spot a difference, an error exists in that line. Jot down the number of lines where contradictions occur, LIST each line, spot the errors, and correct them.

```
•5000 FORX=49152TO49488:READY:POKEX,Y:NEXT:END GJ
•5001 DATA32,161,192,165,43,133,251,165,44,133 DL
•5002 DATA252,160,0,132,254,32,37,193,234,177  DB
•5003 DATA251,208,3,76,138,192,230,251,208,2   OF
•5004 DATA230,252,76,43,192,76,73,78,69,32     KN
•5005 DATA35,32,0,169,35,160,192,32,30,171     CA
•5006 DATA160,0,177,251,170,230,251,208,2,230  CE
•5007 DATA252,177,251,32,205,189,169,58,32,210 JE
•5008 DATA255,169,0,133,253,230,254,32,37,193  CL
•5009 DATA234,165,253,160,0,76,13,193,133,253  NB
•5010 DATA177,251,208,237,165,253,41,240,74,74 MB
•5011 DATA74,74,24,105,65,32,210,255,165,253   EP
•5012 DATA41,15,24,105,65,32,210,255,169,13    GH
•5013 DATA32,220,192,230,63,208,2,230,64,230   AN
•5014 DATA251,208,2,230,252,76,11,192,169,153  NG
•5015 DATA160,192,32,30,171,166,63,165,64,76   BF
•5016 DATA231,192,96,76,73,78,69,83,58,32      EP
•5017 DATA0,169,247,160,192,32,30,171,169,3    PJ
•5018 DATA133,254,32,228,255,201,83,240,6,201  FK
•5019 DATA80,208,245,230,254,32,210,255,169,4  FL
•5020 DATA166,254,160,255,32,186,255,169,0,133 CL
•5021 DATA63,133,64,133,2,32,189,255,32,192    GC
•5022 DATA255,166,254,32,201,255,76,73,193,96  NN
•5023 DATA32,210,255,173,141,2,41,1,208,249    NH
•5024 DATA96,32,205,189,169,13,32,210,255,32   IM
•5025 DATA204,255,169,4,76,195,255,147,83,67   KC
•5026 DATA82,69,69,78,32,79,82,32,80,82        DC
•5027 DATA73,78,84,69,82,32,63,32,0,76         ML
•5028 DATA44,193,234,177,251,201,32,240,6,138  GN
•5029 DATA113,251,69,254,170,138,76,88,192,0   JK
•5030 DATA0,0,0,230,251,208,2,230,252,96       NA
•5031 DATA170,177,251,201,34,208,6,165,2,73    DM
•5032 DATA255,133,2,165,2,208,218,177,251,201  JA
•5033 DATA32,208,212,198,254,76,29,193,0,169   FM
•5034 DATA13,76,210,255,0,0,0                  PA
```

FLANKSPEED FOR THE C-64

By Gordon F. Wheat

Flankspeed will allow you to enter machine language *Ahoy!* programs without any mistakes. Once you have typed the program in, save it for future use. While entering an ML program with *Flankspeed* there is no need to enter spaces or hit the carriage return. This is all done automatically. If you make an error in a line a bell will ring and you will be asked to enter it again. To LOAD in a program Saved with *Flankspeed* use LOAD "name",1,1 for tape, or LOAD "name",8,1 for disk. The function keys may be used after the starting and ending addresses have been entered.

f1—SAVEs what you have entered so far.

f3—LOADs in a program worked on previously.

f5—To continue on a line you stopped on after LOADing in the previously saved work.

f7—Scans through the program to locate a particular line, or to find out where you stopped the last time you entered the program. f7 temporarily freezes the output as well.

```
•5 POKE53280,12:POKE53281,11                   LL
•6 PRINT"[CLEAR][c 8][RVSON][15" "]FLANKSPEED[
 15" "]";                                      ED
•10 PRINT"[RVSON][5" "]MISTAKEPROOF ML ENTRY P
 ROGRAM[6" "]"                                 MC
•15 PRINT"[RVSON][9" "]CREATED BY G. F. WHEAT[
 9" "]"                                        DM
•20 PRINT"[RVSON][3" "]COPR. 1984, ION INTERNA
```

```
TIONAL INC.[3" "]"                                DH
·30 FORA=54272TO54296:POKEA,0:NEXT                IM
·40 POKE54272,4:POKE54273,48:POKE54277,0:POKE5
4278,249:POKE54296,15                             NH
·70 FORA=680TO699:READB:POKEA,B:NEXT              KO
·75 DATA169,251,166,253,164,254,32,216,255,96     HJ
·76 DATA169,0,166,251,164,252,32,213,255,96       JB
·80 B$="STARTING ADDRESS IN HEX":GOSUB2010:AD=
B:SR=B                                            HC
·85 GOSUB2520:IFB=0THEN80                         FO
·86 POKE251,T(4)+T(3)*16:POKE252,T(2)+T(1)*16     KE
·90 B$="ENDING ADDRESS IN HEX":GOSUB2010:EN=B     IF
·95 GOSUB2510:IFB=0THEN80                         FP
·96 POKE254,T(2)+T(1)*16:B=T(4)+1+T(3)*16         MN
·97 IFB>255THENB=B-255:POKE254,PEEK(254)+1        GE
·98 POKE253,B:PRINT                               HN
·100 REM  GET HEX LINE                            IL
·110 GOSUB3010:PRINT": [c P][LEFT]";:FORA=0TO8    FG
·120 FORB=0TO1:GOTO210                            MD
·125 NEXTB                                        ME
·130 A%(A)=T(1)+T(0)*16:IFAD+A-1=ENTHEN310        LH
·135 PRINT" [c P][LEFT]";                         IK
·140 NEXTA:T=AD-(INT(AD/256)*256):PRINT" "        PD
·150 FORA=0TO7:T=T+A%(A):IFT>255THENT=T-255       LK
·160 NEXT                                         IA
·170 IFA%(8)<>TTHENGOSUB1010:GOTO110              FK
·180 FORA=0TO7:POKEAD+A,A%(A):NEXT:AD=AD+8:GOT
O110                                              MN
·200 REM    GET HEX INPUT                         AB
·210 GETA$:IFA$=""THEN210                         HO
·211 IFA$=CHR$(20)THEN270                         GC
·212 IFA$=CHR$(133)THEN4000                       MD
·213 IFA$=CHR$(134)THEN4100                       KF
·214 IFA$=CHR$(135)THENPRINT" ":GOTO4500          GE
·215 IFA$=CHR$(136)THENPRINT" ":GOTO4700          BJ
·220 IFA$>"@"ANDA$<"G"THENT(B)=ASC(A$)-55:GOTO
250                                               GM
·230 IFA$>"/"ANDA$<":"THENT(B)=ASC(A$)-48:GOTO
250                                               LE
·240 GOSUB1100:GOTO210                            LL
·250 PRINTA$"[c P][LEFT]";                        OA
·260 GOTO125                                      CG
·270 IFA>0THEN280                                 OP
·272 A=-1:IFB=1THEN290                            OB
·274 GOTO140                                      CJ
·280 IFB=0THENPRINTCHR$(20);CHR$(20);:A=A-1       HG
·285 A=A-1                                        BE
·290 PRINTCHR$(20);:GOTO140                       KH
·300 REM LAST LINE                                AD
·310 PRINT" ":T=AD-(INT(AD/256)*256)              GJ
·320 FORB=0TOA-1:T=T+A%(B):IFT>255THENT=T-255     PL
·330 NEXT                                         IA
·340 IFA%(A)<>TTHENGOSUB1010:GOTO110              KF
·350 FORB=0TOA-1:POKEAD+B,A%(B):NEXT              HN
·360 PRINT:PRINT"YOU ARE FINISHED!":GOTO4000      ON
·1000 REM BELL AND ERROR MESSAGES                 FL
·1010 PRINT:PRINT"LINE ENTERED INCORRECTLY":PR
INT:GOTO1100                                      DH
·1020 PRINT:PRINT"INPUT A 4 DIGIT HEX VALUE!":
GOTO1100                                          JA
·1030 PRINT:PRINT"ENDING IS LESS THAN STARTING
!":B=0:GOTO1100                                   HD
·1040 PRINT:PRINT"ADDRESS NOT WITHIN SPECIFIED
 RANGE!":B=0:GOTO1100                             AG
·1050 PRINT:PRINT"NOT ZERO PAGE OR ROM!":B=0:G
OTO1100                                           KN
·1060 PRINT"?ERROR IN SAVE":GOTO1100              EI
·1070 PRINT"?ERROR IN LOAD":GOTO1100              GL
·1080 PRINT:PRINT:PRINT"END OF ML AREA":PRINT     PG
·1100 POKE54276,17:POKE54276,16:RETURN            BH
·1200 OPEN15,8,15:INPUT#15,A,A$:CLOSE15:PRINTA
$:RETURN                                          IM
·2000 REM GET FOUR DIGIT HEX                      PC
·2010 PRINT:PRINTB$;:INPUTT$                      GM
·2020 IFLEN(T$)<>4THENGOSUB1020:GOTO2010          II
·2040 FORA=1TO4:A$=MID$(T$,A,1):GOSUB2060:IFT(
A)=16THENGOSUB1020:GOTO2010                       AD
·2050 NEXT:B=(T(1)*4096)+(T(2)*256)+(T(3)*16)+
T(4):RETURN                                       GF
·2060 IFA$>"@"ANDA$<"G"THENT(A)=ASC(A$)-55:RET
URN                                               EH
·2070 IFA$>"/"ANDA$<":"THENT(A)=ASC(A$)-48:RET
URN                                               KP
·2080 T(A)=16:RETURN                              NP
·2500 REM ADRESS CHECK                            LI
·2510 IFAD>ENTHEN1030                             MI
·2515 IFB<SRORB>ENTHEN1040                        MG
·2520 IFB<256OR(B>40960ANDB<49152)ORB>53247THE
N1050                                             MI
·2530 RETURN                                      IM
·3000 REM ADDRESS TO HEX                          EB
·3010 AC=AD:A=4096:GOSUB3070                      HG
·3020 A=256:GOSUB3070                             CE
·3030 A=16:GOSUB3070                              PN
·3040 A=1:GOSUB3070                               MJ
·3060 RETURN                                      IM
·3070 T=INT(AC/A):IFT>9THENA$=CHR$(T+55):GOTO3
090                                               CJ
·3080 A$=CHR$(T+48)                               JP
·3090 PRINTA$;:AC=AC-A*T:RETURN                   AC
·4000 A$="**SAVE**":GOSUB4200                     AI
·4050 OPEN1,T,1,A$:SYS680:CLOSE1                  LH
·4060 IFST=0THENEND                               EO
·4070 GOSUB1060:IFT=8THENGOSUB1200                FJ
·4080 GOTO4000                                    FF
·4100 A$="**LOAD**":GOSUB4200                     AB
·4150 OPEN1,T,0,A$:SYS690:CLOSE1                  MF
·4160 IFST=64THEN110                              JH
·4170 GOSUB1070:IFT=8THENGOSUB1200                CM
·4180 GOTO4100                                    FO
·4200 PRINT" ":PRINTTAB(14)A$                     FG
·4210 PRINT:A$="":INPUT"FILENAME";A$              OM
·4215 IFA$=""THEN4210                             GF
·4220 PRINT:PRINT"TAPE OR DISK?":PRINT            DF
·4230 GETB$:T=1:IFB$="D"THENT=8:A$="@0:"+A$:RE
TURN                                              IG
·4240 IFB$<>"T"THEN4230                           FN
·4250 RETURN                                      IM
·4500 B$="CONTINUE FROM ADDRESS":GOSUB2010:AD=
B                                                 DK
·4510 GOSUB2515:IFB=0THEN4500                     MA
·4520 PRINT:GOTO110                               OI
·4700 B$="BEGIN SCAN AT ADDRESS":GOSUB2010:AD=
B                                                 FH
·4705 GOSUB2515:IFB=0THEN4700                     NK
·4706 PRINT:GOTO4740                              DI
·4710 FORB=0TO7:AC=PEEK(AD+B):GOSUB3030:IFAD+B
=ENTHENAD=SR:GOSUB1080:GOTO110                    BK
·4715 PRINT" ";:NEXTB                             EC
·4720 PRINT:AD=AD+8                               GN
·4730 GETB$:IFB$=CHR$(136)THEN110                 MN
·4740 GOSUB3010:PRINT": ";:GOTO4710               JD
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

PRINT SHOP TO BIT MAP CONVERTOR

FROM PAGE 50

```
•1 REM PRINT SHOP TO BIT MAP CONVERTER      LD
•2 REM FOR PRINT SHOP GRAPHIC EDITOR        BO
•3 REM    *************************         KN
•4 REM    *  (C) MARCH 3, 1984    *         CD
•5 REM    * BY MORTON A. KEVELSON *         DB
•6 REM    * P. O. BOX 260         *         IH
•7 REM    * HOMECREST STATION     *         JG
•8 REM    * BROOKLYN, NY 11229    *         BO
•9 REM    *************************         KN
•10 REM USE WITH DOODLE! FILES              BG
•19 ON FL GOTO200,400                       PN
•20 POKE51,0:POKE52,57:POKE55,0:POKE56,57   OK
•30 PRINT"[RVSON]1[RVSOFF] BIT MAP TO PRI
 NT SHOP"                                   PI
•40 PRINT"[RVSON]2[RVSOFF] PRINT SHOP TO
 BIT MAP"                                   MB
•50 PRINT"PRESS [RVSON]1[RVSOFF] OR [RVSO
 N]2[RVSOFF]"                               OI
•60 GET FL$:FL=VAL(FL$)                     KH
•70 IF FL<1 OR FL>2 THEN GOTO60             MC
•80 INPUT"SOURCE FILE NAME";SF$             OK
•90 INPUT"DESTINATION FILE NAME";DF$        FE
•100 SA=24576:REM SA=DOODL! START ADDRESS   BG
•110 TF=22528:REM PRINT SHOP START ADDRES
 S                                          JG
•120 LOAD SF$,8,1                           BG
•200 FOR RW=0TO7                            OG
•210 FOR PS=0TO7                            OA
•220 FOR CO=0TO10                           BB
•230 BY=SA+CO*8+PS+RW*320                   KB
•240 POKE TF,PEEK(BY)                       PP
•250 TF=TF+1                                LC
•260 NEXTCO:NEXTPS:NEXTRW                   NP
•270 FORI=23100TO23107:POKEI,0:NEXTI        LM
•280 GOSUB800                               CI
•290 POKE43,0:POKE44,88:POKE45,67:POKE46,
 90:SAVE"PSTEMP",8                          AJ
•300 POKE43,1:POKE44,8:POKE45,PEEK(36864)
 :POKE46,PEEK(36865)                        AO
•310 OPEN1,8,15:PRINT#1,"R0:"+DF$+"=PSTEM
 P":CLOSE1:END                              CJ
•400 FOR CM=23552 TO 24551                  JM
•410 POKE CM,1:NEXT CM                      NG
•420 FOR MM=SA TO SA+7999                   EP
•430 POKE MM,0:NEXT MM                      OP
•440 FOR RW=0TO7                            OG
•450 FOR PS=0TO7                            OA
•460 FOR CO=0TO10                           BB
•470 BY=SA+CO*8+PS+RW*320                   KB
•480 POKEBY,PEEK(TF)                        CF
•490 TF=TF+1                                LC
•500 NEXTCO:NEXTPS:NEXTRW                   NP
•510 FORRW=0TO7                             OG
•520 FORPS=0TO7                             OA
•530 BY=SA+88+PS+320*RW                     MC
•540 POKEBY,128                             AO
•550 NEXTPS:NEXTRW                          JB
•560 FORCO=0TO10                            BB
•570 BY=SA+CO*8+1923                        JL
•580 POKEBY,255                             AD
•590 NEXTCO                                 BD
•600 GOSUB800                               CI
•610 POKE43,0:POKE44,92:POKE45,0:POKE46,1
 28                                         HO
•620 SAVE"DDTEMP",8                         AF
•630 POKE43,1:POKE44,8:POKE45,PEEK(36864)
 :POKE46,PEEK(36865)                        AO
•640 OPEN1,8,15:PRINT#1,"R0:"+DF$+"=DDTEM
 P":CLOSE1:END                              MK
•800 POKE36864,PEEK(45)                     EH
•810 POKE36865,PEEK(46)                     IF
•820 RETURN                                 IM
```

SCREEN MAGIC TO DOODLE! CONVERTOR

FROM PAGE 50

```
•10 I=49152                                 AJ
•20 READ A:IF A=256 THEN 40                 FN
•30 POKE I,A:I=I+1:GOTO 20                  OF
•40 PRINT"[CLEAR][DOWN][DOWN][5" "]SCREEN
  MAGIC/DOODLE CONVERTER"                   BF
•50 PRINT"[DOWN][8" "](C) 1984 BY M. BEUT
 JER"                                       KM
•60 PRINT"[3"[DOWN]"]  SELECT :"CHR$(13);
 CHR$(13);"[11" "]1 - DOODLE TO SCREEN MA
 GIC"                                       HO
•70 PRINT"[11" "]2 - SCREEN MAGIC TO DOOD
 LE"                                        LO
•80 INPUT"[DOWN][DOWN][15" "]";A            EE
•90 IFA<1ORA>2THEN40                        IK
•100 IFA=1THEN150                           OL
•110 GOSUB500:SYS49152                      BK
•120 INPUT"ANOTHER CONVERSION";A$           EO
•130 IFLEFT$(A$,1)<>"N"THEN40               II
•140 END                                    IC
•150 GOSUB500:SYS49194                      CE
•160 GOTO120                                CD
•500 PRINT"ENTER THE NAME OF THE EXISTING
  FILE."                                    NB
•505 PRINT"DO NOT INCLUDE THE 'DD' PREFIX
  ON"                                       GK
•506 PRINT"DOODLE FILES!"                   DL
•510 INPUTA$:B=LEN(A$)                      JO
•520 IFA=1ANDB>8THEN510                     LL
•530 IFB>16THEN510                          DO
•540 FORX=1TOB                              KC
•550 POKE49292+X,ASC(MID$(A$,X,1))          HJ
•560 NEXT                                   IA
•570 POKE49289,B                            CA
```

```
•580 B=B+2                                  BA
•590 IFB<17THENPOKE49290,B:RETURN           KF
•600 POKE49290,16                           CK
•610 RETURN                                 IM
•49152 DATA 169,92,133,252,160,0,132,251    GL
•49160 DATA 162,4,169,1,145,251,200,208     CO
•49168 DATA 251,230,252,202,208,246,174,1
 37                                         IB
•49176 DATA 192,160,141,169,96,32,62,192    ME
•49184 DATA 169,92,160,139,174,138,192,76   NN
•49192 DATA 98,192,174,138,192,160,139,16
 9                                          CL
•49200 DATA 92,32,62,192,169,96,174,137     CI
•49208 DATA 192,160,141,76,98,192,142,81    EC
•49216 DATA 192,140,83,192,141,94,192,169   CI
•49224 DATA 1,162,8,160,0,32,186,255        EI
•49232 DATA 169,0,162,0,160,192,32,189      MN
•49240 DATA 255,169,0,162,0,160,0,76        EE
•49248 DATA 213,255,142,116,192,140,118,1
 92                                         JG
•49256 DATA 133,252,169,1,162,8,160,0       JM
•49264 DATA 32,186,255,169,0,162,0,160      EM
•49272 DATA 192,32,189,255,169,0,133,251    BJ
•49280 DATA 169,251,162,255,160,127,76,21
 6                                          EG
•49288 DATA 255,0,0,68,68,256               MC
```

SPEEDING PIXELS

FROM PAGE 37

LISTING 1

```
•1 REM                                      JD
•2 REM      RUPERT REPORT #19               PD
•3 REM      BASIC LISTING 1                 DE
•4 REM      BIT MAP GRAPHICS                KG
•5 REM                                      JD
•6 REM ---DELETE LINES 65 AND 265 IF        JA
•7 REM ---OBJECT FILES CLSCRN.0 AND         MP
•8 REM ---PLOT.$C.0 ARE ON THE DISK         LK
•9 REM                                      JD
•10 ON LL GOTO 75,90                        MJ
•15 DEF FNSB(N)=PEEK(MM) OR 2[UPARROW]N     LG
•20 DEF FNRB(N)=PEEK(MM) AND (255-2[UPARR
OW]N)                                       FP
•30 VV=53248 :REM VIC-II REGISTER 0         FL
•35 :REM >>> PUT BIT MAP AT 8192 <<<
    (SET BIT 3 OF VIC REGISTER 24)          JA
•40 MM=VV+24 : POKE MM,FNSB(3)              OO
•45 :REM >>> SELECT BIT MAP MODE <<<
    (SET BIT 5 OF VIC REGISTER 17)          DH
•50 MM=VV+17 : POKE MM,FNSB(5)              JM
•55 BASE=8192 :REM START BIT MAP MEMORY     KD
•60 :REM >>> CLEAR BIT MAP <<<              HG
•65 FOR M=8000 TO 16191:POKE M,0 :NEXT :
 GOTO 90                                    EP
•70 LL=1 : LOAD"CLSCRN.0",8,1               HC
•75 SYS 828                                 EF
•80 LL=2 : LOAD"PLOT.$C.0",8,1              HN
•85 :REM >>> SELECT COLORS C1 AND C0 <<<    LC
•90 C1=1 : C0=0 : CC=16*C1 + C0             OM
•95 :REM >>FILL SCREEN MEMORY WITH COLOR    AI
•100 FOR MM=1024 TO 2023 :POKE MM,CC:NEXT   OM
•194 :                                      DI
•195 ::REM:: MAIN PROGRAM ::                AE
•196 :                                      DI
•200 PI=3.14159265 : K=PI/40                GE
•205 POKE 198,0  :REM CLEAR KBD BUFFER      KG
•210 Z0=49155:Z1=49156:Z2=49157             EE
•220 FOR Q=5 TO 95 STEP 8                   LA
•230 FOR X=0 TO 319 STEP 6-LOG(Q)           BB
•240 XMSB=0 : XLSB=INT(X)                   ME
•250 IF XLSB>255THENXLSB=XLSB-256:XMSB=1    MB
•260 Y=Q*SIN(X*K)+95                        FL
•265 GOSUB 400 : GOTO 280                   IC
•270 POKE Z0,XLSB:POKE Z1,XMSB:POKE Z2,Y:
 SYS 49152                                  CD
•280 IF PEEK(198)<>0 THEN 300               CO
•290 NEXT:NEXT                              EF
•295 IF PEEK(198)=0 THEN 295 :REM CHECK K
 BD BUFFER                                  DN
•296 :                                      DI
•297 ::REM:: BACK TO NORMAL ::              EL
•298 :                                      DI
•300 :REM >>> RESET BIT MAP MODE <<<        HJ
•310 MM=VV+17 : POKE MM,FNRB(5)             JN
•320 :REM >>> RESTORE SCREEN MEMORY BASE    GC
•330 MM=VV+24 : POKE MM,FNRB(3)             MP
•340 PRINT CHR$(147) :REM CLEAR SCREEN      EG
•390 END                                    IC
•394 :                                      DI
•395 :REM >>> TURN ON PIXEL AT (X,Y)        OB
•396 :REM >>> THIS SUBROUTINE IS REPLACED
  BY THE ML ROUTINE CALLED AT LINE 270      BH
•397 :                                      DI
•400 BIT=7-(X AND 7)                        PJ
•410 MM=BASE + 320*INT(Y/8) + 8*INT(X/8)
  +(Y AND 7)                                AP
•420 POKE MM,FNSB(BIT)                      MO
•430 RETURN                                 IM
```

Listings 2 and 3 are *Flankspeed* versions of the CLRSCR.O and PLOT.$C.S programs listed in assembled format in the *Speeding Pixels* article (on pages 39 and 43 respectively).

LISTING 2

First byte: 033C   Last byte: 0368   Sys to Start: 828

```
033C: 4C 44 03 00 20 1F 40 00 4F
0344: AD 3F 03 85 FB AD 40 03 A6
034C: 85 FC AE 41 03 A0 00 AD 10
0354: 43 03 88 91 FB D0 FB E6 64
035C: FC CA 30 07 D0 F4 AC 42 10
0364: 03 D0 EF 60 00 88
```

LISTING 3

First byte: C000   Last byte: C0AF   Sys to Start: 49152

```
C000: 4C 08 C0 3F 01 00 00 20 75
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering *Ahoy!* programs. Refer to these pages **before** entering any programs!

```
C008: AD 04 C0 F0 13 A9 01 8D B6
C010: 04 C0 AD 03 C0 30 04 C9 44
C018: 40 30 05 A9 3F 8D 03 C0 C7
C020: AD 05 C0 10 0B 29 7F C9 21
C028: 48 30 05 A5 C7 8D 05 C0 66
C030: AD 05 C0 29 F8 85 FB A9 F0
C038: 00 85 FC A5 FB 18 2A 26 C4
C040: FC 2A 26 FC 2A 26 FC 85 5D
C048: FD A5 FC 85 FE A5 FD 2A 3B
C050: 26 FC 2A 26 FC 18 65 FD 3C
C058: 85 FD A5 FC 65 FE 85 FE 67
C060: AD 05 C0 29 07 18 65 FD 7F
C068: 85 FD A5 FE 69 00 85 FE 7E
C070: AD 03 C0 29 F8 18 65 FD 7F
C078: 85 FD AD 04 C0 65 FE 85 58
C080: FE AD 06 C0 18 65 FD 85 F4
C088: FD AD 07 C0 65 FE 85 FE E4
C090: AD 03 C0 29 07 85 FB A9 5D
C098: 07 38 E5 FB AA 38 A9 00 46
C0A0: 2A CA 10 FC 85 FC A0 00 C5
C0A8: B1 FD 05 FC 91 FD 60 FF 4A
```

Trio Allegretto

FROM PAGE 18 SONG MAKER

```
•1 REM "SONG MAKER"                                      IL
•3 REM THIS PROGRAM LETS YOU COPY THE FR$
 () STRINGS FROM THE SCREEN DISPLAY                      KI
•5 REM OR SAVE THE FR$() STRINGS ON DISK
 FOR RETRIEVAL BY "SONG LOADER"                          EN
•7 REM YOU TYPE IN SONGS AT 9500-9890 AND
 NAME THE SONG SAVE FILE AT 200                          KB
•10 GOSUB 9000:GOTO 80                                   IB
•60 FOR N=1 TO LEN(MD$(PH)):SYS MM                       LH
•62 F$=FR$(N,PH)                                         DG
•65 SYS M                                                ON
•66 IF DS=1 THEN GOSUB 500                               OE
•67 FOR I=0 TO DU%(VAL(MID$(MD$(PH),N,1))
 ):NEXT                                                  JB
•68 REM SYS MM:REM STACCATO NOTES                        GA
•69 NEXT:PH=PH+1:IF PH>ES% THEN PH=0                     IJ
•70 SYS MM:RETURN                                        AO
•80 PRINT "[CLEAR][13"[DOWN]"]SHIFT = 'PL
 AY NEXT PHRASE'"                                        KA
•81 PRINT "[4" "]Q = 'QUIT'"                             LA
•82 PRINT "[4" "]S = 'SAVE SONG ON DISK'"                JM
•83 PRINT "[4" "]D = 'MAKE DATA STATEMENT
 S AT LINE[12" "]9400'"                                  GA
•84 PRINT "[4" "]P = 'PRINT EACH F$ STRIN
 G AS IT"                                                GG
•85 PRINT "[9" "]PLAYS; WAIT FOR KEYPRESS
 "                                                       OK
•86 PRINT "[9" "]BETWEEN NOTES'"                         NN
•87 PRINT "[4" "]N = 'NO PRINTING OF F$ S
 TRINGS'"                                                GJ
•90 DS=0                                                 HO
•100 PRINT "[HOME][16" "][HOME]PHRASE "PH                DD
•110 IF PEEK (653)<>0 THEN GOSUB 60:GOTO
 100                                                     ND
•120 A=PEEK(203):IF A=64 THEN 110                        PK
•130 IF A=62 THEN POKE 198,0:SYS 65126:RE
 M "WARM START" ENDS PROGRAM                             IC
•135 IF A=41 THEN DS=1:GOTO 110:REM SET F
 LAG TO PRINT FR$() VALUES                               OJ
•140 IF A=13 THEN PRINT "[HOME]SAVING DIS
 K FILE":GOSUB 200                                       GD
•145 IF A=39 THEN DS=0:REM SET FLAG TO ST
 OP PRINTING FR$()                                       OO
•150 IF A=18 THEN 20000:REM MAKE DATA STA
 TEMENTS AND WIPE OUT LINES>9400                         FG
•190 GOTO 110                                            CC
•200 OPEN 2,8,2,"@0:SONG #1,S,W"                         MO
•210 CR$=CHR$(13):PRINT#2,ES%CR$EV%CR$LD%
 CR$;                                                    PI
•215 FOR I=0 TO 2:PRINT#2,G%(I)CR$;:NEXT:
 FOR I=0 TO 2                                            NF
•220 PRINT#2,AK%(I)CR$DY%(I)CR$SN%(I)CR$R
 E%(I)CR$;                                               PL
•225 FOR N=0 TO 1:PRINT#2,PW%(I,N)CR$;:NE
 XT:NEXT                                                 BA
•230 FOR I=0 TO 9:PRINT#2,DU%(I)CR$;:NEXT                PH
•235 FOR PH=0 TO ES%:PRINT#2,MD$(PH)CR$;:
 PRINT "<";                                              KM
•240 FOR N=1 TO LEN(MD$(PH)):FOR I=1 TO 6                PF
•245 PRINT#2,MID$(FR$(N,PH),I,1)CR$;:NEXT
 :NEXT:NEXT                                              DM
•290 CLOSE 2:PH=0:RETURN                                 FI
•500 PRINT "[HOME][DOWN][DOWN]NOTE[5" "][
 3"[LEFT]"]"N:FOR I=0 TO 2:PRINT "[20" "]
 ":NEXT                                                  LJ
•505 PRINT "[HOME][3"[DOWN]"]":FOR I=1 TO
 5 STEP 2                                                MJ
•507 PRINT "[15" "][15"[LEFT]"]";                        GA
•510 PRINT ASC(MID$(F$,I,1))"   "ASC(MID$(
 F$,I+1,1))"   ":NEXT                                    GO
•515 PRINT "PRESS SHIFT TO GO ON"                        CB
•520 IF PEEK(653)=0 THEN 520                             HO
•525 PRINT "[UP][21" "]"                                 BL
•530 RETURN                                              IM
•8998 REM SET UP SOUND SHAPE                             FD
•9000 F$="F":I=0:N=0:VC=0:M=848:MM=823:RE
 M M&MM PUT ML IN CASSETTE BUFFER                        JG
•9001 PH=0:ES%=8:DIM MD$(ES%),G%(2),AD(2)
 ,Y%(2)                                                  MK
•9002 DIM ME$(ES%,2),MV$(ES%,2)                          ED
•9003 REM PREVIOUS LINE:9002 DIM ME$(ES%,
 2),MV$(ES%,2)                                           GE
•9004 DIM DU%(9),PI%(168,1),AK%(2),DY%(2)
 ,SN%(2),RE%(2),WF%(2),PW%(2,1)                          IP
•9008 REM ATTACK--VOICES 0,1,2                           KG
•9009 REM NUMBER FROM 0 TO 15; LOWER NUMB
 ER=SHARPER ATTACK                                       OK
```

```
•9010 AK%(0)=0:AK%(1)=0:AK%(2)=0             CP
•9015 FOR I=0 TO 2:AK%(I)=AK%(I)*16:NEXT     FF
•9018 REM DECAY--VOICES 0,1,2                PK
•9019 REM NUMBER FROM 0 TO 15; LOWER NUMB
 ER=FASTER DECLINE                           GA
•9020 DY%(0)=5:DY%(1)=3:DY%(2)=3             AL
•9028 REM SUSTAIN--VOICES 0,1,2              FC
•9029 REM NUMBER FROM 0 TO 15; LOWER NUMB
 ER=SOFTER VOLUME DURING SUSTAIN             JM
•9030 SN%(0)=2:SN%(1)=0:SN%(2)=0             CI
•9035 FOR I=0 TO 2:SN%(I)=SN%(I)*16:NEXT     BD
•9038 REM RELEASE--VOICES 0,1,2              OK
•9039 REM NUMBER FROM 0 TO 15; LOWER NUMB
 ER=FASTER DROP TO SILENCE AT END            PJ
•9040 RE%(0)=0:RE%(1)=3:RE%(2)=5             BG
•9048 REM SET SOUND ADDRESSES                JC
•9050 FOR I=0 TO 2:AD(I)=54277+7*I:NEXT      IL
•9058 REM POKE ADSR ENVELOPES                JL
•9060 FOR I=0 TO 2:POKE AD(I),AK%(I) OR D
 Y%(I)                                       DM
•9065 POKE AD(I)+1,SN%(I) OR RE%(I):NEXT     DK
•9067 REM SET UP GATES                       JN
•9068 REM WAVEFORMS, VOICES 0,1,2 (ADD VA
 LUES):                                      DJ
•9069 REM TRIANGLE ON=16; SAWTOOTH ON=32;
  PULSE ON=64 (SET WIDTH!); NOISE ON=128     IL
•9070 WF%(0)=32:WF%(1)=64:WF%(2)=64          LH
•9075 G%(0)=WF%(0)OR 1:G%(1)=WF%(1) OR 3:
 G%(2)=WF%(2) OR 1                           CG
•9078 REM SET PULSE WIDTHS                   AC
•9079 REM VOICES 0,1,2; LOW BYTE, HIGH BY
 TE                                          MG
•9080 PW%(0,0)=200:PW%(0,1)=3                PB
•9081 PW%(1,0)=200:PW%(1,1)=10               HJ
•9082 PW%(2,0)=200:PW%(2,1)=7                KB
•9085 FOR I=0 TO 2:VC=54274+I*7:FOR N=0 T
 O 1                                         OB
•9086 POKE VC+N,PW%(I,N):NEXT:NEXT           NG
•9098 REM SET UP DURATIONS                   ME
•9100 FOR I=0 TO 9:READ DU%(I):NEXT          MB
•9105 DATA 40,96,128,192,256,384,512,640,
 768,1024                                    EG
•9196 REM MACHINE LANGUAGE ROUTINE (AT M)    HM
•9198 REM FIND ADDRESS OF F$ AND PUT IT I
 N ZERO PAGE AT 139, 140                     MA
•9200 N=PEEK(45)+256*PEEK(46)+3:Y%=N/256:
 X%=N-Y%*256                                 IF
•9205 POKE M,173:POKE M+1,X%:POKE M+2,Y%:
 N=N+1:Y%=N/256:X%=N-Y%*256                  AN
•9210 POKE M+3,133:POKE M+4,251:POKE M+5,
 173:POKE M+6,X%:POKE M+7,Y%                 NK
•9215 POKE M+8,133:POKE M+9,252:POKE M+10
 ,160:POKE M+11,0                            HA
•9220 POKE M+12,162:POKE M+13,0              GG
•9223 REM GET EACH PITCH FROM F$ AND PUT
 IT IN FREQUENCY REGISTER                    CH
•9225 FOR I=M+14 TO M+54 STEP 8:POKE I,17
 7:POKE I+1,251                              KF
•9230 POKE I+2,157:POKE I+3,0:POKE I+4,21
 2                                           HK
•9235 POKE I+5,200:POKE I+6,162:READ A:PO
 KE I+7,A:NEXT                               IE
•9236 DATA 1,7,8,14,15,4                     AK
•9238 REM GATE EACH SOUND OPEN               MC
•9240 N=0:FOR I=M+59 TO M+71 STEP 6          BJ
•9241 POKE I,173:A=MM-3+N:Y%=A/256:X%=A-2
 56*Y%:POKE I+1,X%:POKE I+2,Y%               PO
•9242 POKE I+3,141:READ A:POKE I+4,A:POKE
  I+5,212:N=N+1:NEXT                         DG
•9243 DATA 4,11,18                           PB
•9248 REM GARBAGE COLLECTION                 ME
•9250 FOR I=M+77 TO M+90:READ A:POKE I,A:
 NEXT                                        CE
•9255 DATA 164,52,165,51,105,6,144,1,200,
 133,51,132,52,96                            BN
•9258 REM GATE-OFF ML ROUTINE AT MM          LA
•9260 FOR I=MM TO MM+16 STEP 8               GG
•9261 POKE I,173:Y%=3:X%=34+(I-MM)/8:POKE
  I+1,X%:POKE I+2,Y%:REM USES 820-822        BB
•9262 POKE I+3,41:POKE I+4,254               DI
•9263 POKE I+5,141:READ A:POKE I+6,A:POKE
  I+7,212:NEXT:POKE MM+24,96                 KD
•9264 DATA 4,11,18                           PB
•9268 REM SET WAVEFORMS WITH GATES OFF       NK
•9269 REM STORED AT 820-822--LINE  9261 R
 EQUIRES THIS (820=HEX 03 34)                BO
•9270 POKE 820,G%(0):POKE 821,G%(1):POKE
 822,G%(2):SYS MM                            OB
•9297 REM SET FILTER AND VOLUME              BN
•9298 REM FILTER FREQUENCY                   DE
•9299 REM LOW BYTE (0-7) X%; HIGH BYTE (0
 -255) Y%                                    PA
•9300 X%=3:Y%=150                            GF
•9305 POKE 54293,X%:POKE 54294,Y%            KJ
•9308 REM FILTER ON?                         DD
•9309 REM VOICE 1 ON=1; 2 ON=2; 3 ON=4; 1
 &2 ON=3; 2&3 ON=6; ALL ON=7                 AL
•9310 X%=0                                   FM
•9318 REM FILTER RESONANCE                   CE
•9319 REM PEAK VOLUME (0=LOW, 15=HIGH)       FH
•9320 Y%=14                                  JL
•9325 Y%=Y%*16:POKE 54295,X% OR Y%           MG
•9328 REM SELECT FILTER TYPE                 CP
•9329 REM LOW-PASS=1;BAND-PASS=2;HIGH-PAS
 S=4;LO-BAND=3;HI-BAND=6;ALL=7               MH
•9330 X%=1                                   FP
•9335 X%=X%*16                               LC
•9338 REM SELECT OVERALL VOLUME              FJ
•9339 REM 15=HIGH, 0=LOW                     PB
•9340 Y%=15:POKE 54296,X% OR Y%              EK
•9345 EV%=2:REM SET NUMBER OF VOICES (MIN
 US 1)                                       IN
•9350 GOTO 9400                              GG
•9358 REM MUSIC DATA LOADER (EFFECTIVE ON
 LY IF 'MAKE DATA' WAS EXECUTED              NP
•9359 REM DURING MAIN LOOP)                  PB
```

```
·9360 READ ES%:READ LD%:DIM FR$(LD%,ES%) NA
·9365 FOR PH=0 TO ES%:READ MD$(PH):FOR N=
 1 TO LEN(MD$(PH)):PRINT "!";           MG
·9370 FOR I=1 TO 6:READ A:FR$(N,PH)=FR$(N
 ,PH)+CHR$(A):NEXT:NEXT:NEXT            AD
·9375 PH=0:RETURN                       DO
·9397 REM SET UP PITCH ARRAY            DK
·9398 REM EACH NOTE, IN ALL ITS OCTAVES JB
·9399 REM C                             NE
·9400 X%=3:GOSUB 9490                   CB
·9401 DATA 12,1,24,2,48,4,97,8,195,16,135
 ,33,15,67,30,134                       IF
·9402 REM D                             NH
·9403 X%=4:GOSUB 9490                   PG
·9404 DATA 45,1,90,2,180,4,104,9,209,18,1
 62,37,69,75,139,150                    PA
·9405 REM E (F-FLAT)                    MH
·9406 X%=5:GOSUB 9490:Y%=13:GOSUB 9495  IL
·9407 DATA 81,1,163,2,71,5,143,10,31,21,6
 2,42,125,84,250,168                    HM
·9408 REM F (E-SHARP)                   DB
·9409 X%=6:GOSUB 9490:Y%=19:GOSUB 9495  IK
·9410 DATA 102,1,204,2,152,5,48,11,96,22,
 193,44,131,89,6,179                    IL
·9411 REM G                             NI
·9412 X%=7:GOSUB 9490                   PF
·9413 DATA 145,1,35,3,71,6,143,12,30,25,6
 0,50,121,100,243,200                   GM
·9414 REM A                             NC
·9415 X%=1:GOSUB 9490                   OH
·9416 DATA 195,1,134,3,12,7,24,14,49,28,9
 9,56,199,112,143,225                   PI
·9417 REM B                             NF
·9418 X%=2:GOSUB 9490                   PE
·9419 DATA 250,1,244,3,233,7,210,15,165,3
 1,75,63,151,126,46,253                 CF
·9420 REM D-FLAT (C-SHARP)              JL
·9421 X%=11:GOSUB 9490:Y%=17:GOSUB 9495 NH
·9422 DATA 28,1,56,2,112,4,225,8,195,17,1
 34,35,12,71,24,142                     KJ
·9423 REM E-FLAT (D-SHARP)              FD
·9424 X%=12:GOSUB 9490:Y%=18:GOSUB 9495 GF
·9425 DATA 62,1,125,2,251,4,247,9,239,19,
 223,39,191,79,126,159                  FB
·9426 REM G-FLAT (F-SHARP)              HH
·9427 X%=14:GOSUB 9490:Y%=20:GOSUB 9495 IM
·9428 DATA 123,1,246,2,237,5,218,11,181,2
 3,107,47,214,94,172,189                HG
·9429 REM A-FLAT (G-SHARP)              HE
·9430 X%=8:GOSUB 9490:Y%=21:GOSUB 9495  LD
·9431 DATA 169,1,83,3,167,6,78,13,156,26,
 57,53,115,106,230,212                  PA
·9432 REM B-FLAT (A-SHARP)              FP
·9433 X%=9:GOSUB 9490:Y%=15:GOSUB 9495  GB
·9434 DATA 221,1,187,3,119,7,239,14,223,2
 9,190,59,124,119,248,238               CN
·9435 REM C-FLAT                        CH
·9436 X%=10:GOSUB 9490                  EO
·9437 DATA 4,1,250,1,244,3,233,7,210,15,1
 65,31,75,63,151,126                    AK
·9438 REM B-SHARP                       IN
·9439 X%=16:GOSUB 9490                  CE
·9440 DATA 24,2,48,4,97,8,195,16,135,33,1
 5,67,30,134,255,255                    NM
·9485 GOTO 9500                         FP
·9489 REM READ PITCHES                  NA
·9490 FOR I=0 TO 147 STEP 21:READ PI%(I+X
 %,0),PI%(I+X%,1):NEXT:RETURN           FN
·9494 REM IDENTICAL PITCHES             GA
·9495 FOR I=0 TO 147 STEP 21:PI%(I+Y%,0)=
 PI%(I+X%,0):PI%(I+Y%,1)=PI%(I+X%,1)    LN
·9496 NEXT:RETURN                       EJ
·9497 REM EACH PHRASE HAS ONLY ONE DU%(PH
 ) STRING, NO MATTER HOW MANY VOICES    AO
·9498 REM EACH PHRASE HAS ONE ME$(PH,VC)
 & ONE MV$(PH,VC) STRING PER VOICE      IP
·9499 REM PHRASE 0                      IP
·9500 MD$(0) = "100[7"1"]00[6"1"]"      CL
·9501 ME$(0,0)="[4"F"]GAFG@[4"F"]GAFGE" FG
·9502 MV$(0,0)="6[17" "]"               PD
·9503 ME$(0,1)="CC@DCGD[4"C"]@DCGDCC"   EG
·9504 MV$(0,1)="45 45354545 453545"     BJ
·9505 ME$(0,2)="FA@EAGBC[c B]FA@DAGBC[c B
 ]"                                     CA
·9506 MV$(0,2)="34 34343434 343434"     EI
·9509 REM PHRASE 1                      IO
·9510 MD$(1) = "100[14"1"]"             PO
·9511 ME$(1,0)="[4"F"]GAFGEF[7"@"]"     JP
·9512 MV$(1,0)="6[16" "]"               OE
·9513 ME$(1,1)="FC@ECDBCCFCECDACA"      FD
·9514 MV$(1,1)="45 454  545454  5"      BO
·9515 ME$(1,2)="FA@EADBC[c B]FAEADFCF"  II
·9516 MV$(1,2)="34 34343434343435"      OI
·9519 REM PHRASE 2                      JB
·9520 MD$(2) = "100[7"1"]00[6"1"]"      JF
·9521 ME$(2,0)="[c B]@[c B][c B]CD[c B]C@
 [c B]@[c B][c B]CD[c B]CA"             BA
·9522 MV$(2,0)="6[3" "]7 67 6[3" "]7 676" IH
·9523 ME$(2,1)="@F@@FCGFF@F@@FCGFF"     DA
·9524 MV$(2,1)=" 5[3" "]4545[5" "]4545" PB
·9525 ME$(2,2)="[c B]D@GDCEF[c E][c B]D@G
 DCEF[c E]"                             HD
·9526 MV$(2,2)="35 35353535 353535"     HE
·9529 REM PHRASE 3                      JA
·9530 MD$(3) = "100[12"1"]"             DH
·9531 ME$(3,0)="[c B]@[c B][c B]CD[c B]CA
 [c B][5"@"]"                           AG
·9532 MV$(3,0)="6[3" "]7 676[6" "]"     OJ
·9533 ME$(3,1)="[c B]F@AFGEFF[c B]FAFGF" JI
·9534 MV$(3,1)="45 454545454545"        LI
·9535 ME$(3,2)="[c B]D@ADGCF[c E][c B]DAD
 GD"                                    BA
·9536 MV$(3,2)="35 353535353535"        MM
·9539 REM PHRASE 4                      JD
·9540 MD$(4) = "[16"1"]"                HA
·9541 ME$(4,0)="F[c B]C@C[3"@"]D[c B]C[5"
```

```
@"]"                                          PF
•9542 MV$(4,0)="6 7[6" "]67[5" "]"            CF
•9543 ME$(4,1)="[3"F"]AG[c B]AADGFAG[c B]
 AA"                                          NK
•9544 MV$(4,1)="45454545 4 54545"             LH
•9545 ME$(4,2)="FDFFGEAF[c B][3"F"]GEAF"      JH
•9546 MV$(4,2)="3535353534353535"             DK
•9549 REM PHRASE 5                            JC
•9550 MD$(5) = "[18"1"]"                      PO
•9551 ME$(5,0)="CFG@G@@GAFG[7"@"]"            EC
•9552 MV$(5,0)="6[17" "]"                     OG
•9553 ME$(5,1)="@BGEGFGE@BFECCAFGC"           MA
•9554 MV$(5,1)=" 4 54545 4 5453[3" "]"        NN
•9555 ME$(5,2)="@AECDBECGACFC[c B]AFGF"       BI
•9556 MV$(5,2)=" 4 54  534343 2  1"           MA
•9559 REM PHRASE 6                            JF
•9560 MD$(6) = "100[7"1"]00[6"1"]"            HJ
•9561 ME$(6,0)="[4"F"]GAFG@[4"F"]GAFGE"       GI
•9562 MV$(6,0)="6[17" "]"                     BF
•9563 ME$(6,1)="CC@DCGD[4"C"]@DCGDCC"         DA
•9564 MV$(6,1)="45 45354545 453545"           KD
•9565 ME$(6,2)="FA@EAGBC[c B]FA@DAGBC[c B
 ]"                                           IK
•9566 MV$(6,2)="34 34343434 343434"           NC
•9569 REM PHRASE 7                            JE
•9570 MD$(7) = "100[13"1"]"                   EP
•9571 ME$(7,0)="F@FFGAFGEC[3"@"]C@@"          BB
•9572 MV$(7,0)="6[12" "]7  "                  MJ
•9573 ME$(7,1)="FC@ECDBCC@A@[c B]CF@"         GE
•9574 MV$(7,1)="45 454  5 4  5  "             EL
•9575 ME$(7,2)="FA@EADBC[c B]FFGRA[c E]@"     NN
•9576 MV$(7,2)="34 343434343435 "             BI
•9579 REM PHRASE 8                            JH
•9580 MD$(8) = "[4"1"]00[12"1"]"              GF
•9581 ME$(8,0)="DC@C@[c B]AFGDF[3"@"]F[3"
 @"]"                                         OA
•9582 MV$(8,0)="7[4" "]6[8" "]7[3" "]"        EN
•9583 ME$(8,1)="[c B]CACA@@C@CFDC[c B]A@F
 @"                                           FG
•9584 MV$(8,1)="5[9" "]46 5  4 "              II
•9585 ME$(8,2)="DCFCF@CAC[c B]F[c B]AGF@F
 @"                                           FG
•9586 MV$(8,2)="54535 4 3435[4" "]3 "         LP
•9898 REM SET UP CONVERSION TABLE             KO
•9900 DIM TB%(255)                            EG
•9910 FOR I=0 TO 255:TB%(I)=0:NEXT            CE
•9911 FOR I=65 TO 72:TB%(I)=I-64:NEXT         NB
•9912 FOR I=193 TO 199:TB%(I)=I-178:NEXT      EN
•9913 FOR I=8 TO 14:READ A:TB%(A)=I:NEXT      AD
•9914 REM NEXT LINE SAYS: 9915 DATA 176,1
 91,188,172,177,187,165                       EB
•9915 DATA 176,191,188,172,177,187,165        CB
•9916 REM (LINE 9915 IS DELETED BY 'MAKE
 DATA' OPTION)                                LD
•9919 REM FIND LONGEST PHRASE, AND DIM FR
 $ TO EXACT LENGTH NEEDED                     DM
•9920 LD%=0:FOR I=0 TO ES%:X%=LEN(MD$(I))
 :IF X%>LD% THEN LD%=X%                       FJ
```

```
•9921 NEXT                                    IA
•9925 DIM FR$(LD%,ES%)                        CN
•9928 REM CONVERT STRINGS TO USABLE FORM      IG
•9930 FOR PH=0 TO ES%:FOR N=1 TO LEN(MD$(
 PH)):FR$(N,PH)=""                            BP
•9935 FOR VC=0 TO EV%:X%=TB%(ASC(MID$(ME$
 (PH,VC),N,1)))                               NG
•9940 V$=MID$(MV$(PH,VC),N,1):IF V$<>" "
 THEN Y%(VC)=21*VAL(V$)                       JN
•9945 IF X%<>0 THEN X%=X%+Y%(VC)              DA
•9950 FR$(N,PH)=FR$(N,PH)+CHR$(PI%(X%,0))
 :FR$(N,PH)=FR$(N,PH)+CHR$(PI%(X%,1))         DP
•9955 NEXT                                    IA
•9960 IF EV%<2 THEN FR$(N,PH)=FR$(N,PH)+L
 EFT$(FR$(N,PH),2)                            AL
•9965 IF EV%<1 THEN FR$(N,PH)=FR$(N,PH)+L
 EFT$(FR$(N,PH),2)                            FK
•9970 PRINT ".";:NEXT:NEXT                    NI
•9980 PH=0                                    HP
•9990 RETURN                                  IM
•19998 REM ROUTINE TO MAKE DATA STATEMENT
 S THROUGH FORCED SCREEN READS                JC
•20000 PRINT "[CLEAR][3"[DOWN]"]MAKING DA
 TA STATEMENTS WIPES OUT ALL THE"             MI
•20001 PRINT "LINES NOT DIRECTLY NEEDED."
 :PRINT:PRINT"[7" "]PROCEED? (Y OR N)"        LM
•20002 A=PEEK(203):IF (A<>39) AND (A<>25)
  THEN 20002                                  AC
•20003 IF A=39 THEN 80                        PI
•20005 A=49152                                PB
•20006 FOR PH=0 TO ES%:B=LEN(MD$(PH)):POK
 E A,B:A=A+1:PRINT ">";:FOR N=1 TO B          KD
•20007 POKE A,ASC(MID$(MD$(PH),N,1)):A=A+
 1:NEXT                                       OF
•20008 FOR N=1 TO B:FOR I=1 TO 6:POKE A,A
 SC(MID$(FR$(N,PH),I,1)):A=A+1                HN
•20009 NEXT:NEXT:NEXT:B=A-1:A=49152:C=940
 0:CR$=CHR$(13):D=4                           CJ
•20010 PRINT "[CLEAR][DOWN][DOWN]9350 REM
  DELETED"CR$"9390 DATA"ES%"[LEFT],"LD%CR
 $;                                           IG
•20011 PRINT "9002 REM DELETED"CR$;           MA
•20015 GOSUB 20085:IF D>8 THEN 20091          GJ
•20020 GOSUB 20080:IF D>8 THEN 20090          CB
•20025 IF A>=B THEN 20092                     BE
•20030 IF N=0 THEN 20015                      GE
•20035 GOTO 20020                             HN
•20040 PRINT "[CLEAR][DOWN][DOWN]";:D=0:C
 R$=CHR$(13):GOTO 20025                       LF
•20041 PRINT "[CLEAR][DOWN][DOWN]";:D=0:C
 R$=CHR$(13):GOTO 20020                       KO
•20079 REM ROUTINE TO PRINT FR$() DATA ST
 ATEMENT ON SCREEN                            DG
•20080 PRINTC"DATA";:FORI=1TO 6:PRINTPEEK
 (A)"[LEFT],";:A=A+1:NEXT:C=C+1:D=D+1:N=N
 -1                                           JI
•20081 PRINT CHR$(20)CR$;:RETURN              OK
•20084 REM ROUTINE TO PRINT MD$() DATA ST
```

```
ATEMENT ON SCREEN                         IJ
•20085 A$="":N=PEEK(A):A=A+1:FOR I=1 TO N
:A$=A$+CHR$(PEEK(A)):A=A+1:NEXT           GB
•20086 PRINT C"DATA"CHR$(34)A$CHR$(34)CR$
;:C=C+1:D=D+1:RETURN                      CD
•20088 REM SET UP LAST LINE TO EXECUTE FR
OM SCREEN                                 BM
•20090 PRINT "A="A"[LEFT]:B="B"[LEFT]:C="
C"[LEFT]:N="N"[LEFT]:GOTO 20040"CR$;:GOT
O 20095                                   NP
•20091 PRINT "A="A"[LEFT]:B="B"[LEFT]:C="
C"[LEFT]:N="N"[LEFT]:GOTO 20041"CR$;:GOT
O 20095                                   FG
•20092 PRINT "C="C-1":GOTO 20100"CR$;:GOT
O 20095                                   ON
•20094 REM LOAD KEYBOARD BUFFER WITH CARR
IAGE RETURNS AND GO READ SCREEN           LO
•20095 FOR I=631 TO 640:POKE I,13:NEXT:PO
KE 198,10:PRINT "[HOME]";:END             OE
•20100 A=PEEK(43)+256*PEEK(44):PRINT "[CL
EAR]";                                    BN
•20105 B=PEEK(A+2)+256*PEEK(A+3):A=PEEK(A
)+256*PEEK(A+1)                           EB
•20106 PRINT "[HOME]"A"  "B"  "C          LA
•20110 IF B=C THEN 20120                  GK
•20115 GOTO 20105                         IH
•20120 POKE A,0:POKE A+1,0:A=A+2          OH
•20125 B=INT(A/256):C=A-256*B             FE
•20130 PRINT "[CLEAR][DOWN][DOWN]150":PRI
NT "83"                                   IJ
•20135 PRINT "POKE45,"C"[LEFT]:POKE46,"B"
[LEFT]:POKE47,"C"[LEFT]:POKE48,"B"[LEFT]
:POKE49,"C                                EH
•20140 PRINT "POKE50,"B"[LEFT]:GOTO 10"   NJ
•20145 FOR I=631 TO 640:POKE I,13:NEXT:PO
KE 198,10:PRINT "[HOME]";:END             OE
```

SONG LOADER

```
•1 REM "SONG LOADER"                      NM
•3 REM THIS IS AN EXAMPLE OF HOW TO LOAD
AND USE A SONG STORED ON DISK             JB
•5 REM USING THE PROGRAM "SONG MAKER"     PD
•6 REM                                    JD
•7 REM LINE 10 AND LINE 9000 MUST EXECUTE
 BEFORE ANYTHING ELSE IN YOUR PROGRAM     FD
•9 REM                                    JD
•10 GOSUB 9000:GOTO 80                    IB
•60 FOR PH=0 TO ES%:FOR N=1 TO LEN(MD$(PH
)):SYS MM                                 MC
•62 F$=FR$(N,PH)                          DG
•65 SYS M                                 ON
•67 FOR I=0 TO DU%(VAL(MID$(MD$(PH),N,1))
):NEXT                                    JB
•68 REM SYS MM:REM STACCATO NOTES         GA
•69 NEXT:NEXT                             EF
•70 SYS MM:RETURN                         AO
•80 PRINT "[CLEAR]WHEN SONG ENDS, PRESS":
PRINT:PRINT "SHIFT TO REPEAT SONG"        PC
•85 PRINT "OR Q TO QUIT"                  FP
•100 IF PEEK(653)<>0 THEN GOSUB 60:GOTO 1
00                                        ND
•110 IF PEEK(203)=62 THEN POKE 198,0:SYS
65126                                     HG
•190 GOTO 100                             CF
•8997 REM                                 JD
•8998 REM SET UP SOUND AND LOAD SONG      MN
•8999 REM                                 JD
•9000 F$="F":I=0:N=0:VC=0:M=848:MM=823:RE
M M & MM PUT ML IN CASSETTE BUFFER        JG
•9001 PH=0:DIM G%(2),AD(2),DU%(9),AK%(2),
DY%(2),SN%(2),RE%(2),WF%(2),PW%(2,1)      DI
•9007 REM                                 JD
•9008 REM OPEN SONG FILE & GET # OF PHRAS
ES AND # OF VOICES                        JJ
•9009 REM                                 JD
•9010 OPEN 2,8,2,"0:SONG #1,S,R"          OM
•9015 INPUT#2,ES%,EV%,LD%                 JC
•9017 REM                                 JD
•9018 REM DIMENSION KEY VARIABLES         NH
•9019 REM                                 JD
•9020 DIM FR$(LD%,ES%),MD$(ES%)           CC
•9027 REM                                 JD
•9028 REM LOAD THE REMAINDER OF THE VARIA
BLES FROM THE SONG FILE                   GI
•9029 REM                                 JD
•9030 FOR I=0 TO 2:INPUT#2,G%(I):NEXT     JO
•9031 FOR I=0 TO 2:INPUT#2,AK%(I),DY%(I),
SN%(I),RE%(I)                             CB
•9032 FOR N=0 TO 1:INPUT#2,PW%(I,N):NEXT:
NEXT                                      JM
•9033 FOR I=0 TO 9:INPUT#2,DU%(I):NEXT    MP
•9034 FOR PH=0 TO ES%:INPUT#2,MD$(PH):FOR
 N=1 TO LEN(MD$(PH))                      BL
•9035 FOR I=1 TO 6:GET#2,A$,B$            PJ
•9036 IF A$="" THEN A$=CHR$(0)            OE
•9037 FR$(N,PH)=FR$(N,PH)+A$:NEXT:NEXT:NE
XT                                        JE
•9040 CLOSE 2                             NC
•9041 REM                                 JD
•9042 REM SET UP ENVELOPES                BM
•9043 REM                                 JD
•9044 FOR I=0 TO 2:AD(I)=54277+7*I:NEXT   IL
•9045 FOR I=0 TO 2:POKE AD(I),AK%(I) OR D
Y%(I)                                     DM
•9046 POKE AD(I)+1,SN%(I) OR RE%(I):NEXT  DK
•9047 REM                                 JD
•9048 REM SET PULSE WIDTHS                AC
•9049 REM                                 JD
•9050 FOR I=0 TO 2:VC=54274+I*7:FOR N=0 T
O 1                                       OB
•9051 POKE VC+N,PW%(I,N):NEXT:NEXT        NG
•9066 REM                                 JD
•9067 REM SET FILTER AND VOLUME           BN
•9068 REM FILTER FREQUENCY                DE
•9069 REM LOW BYTE (0-7) X%; HIGH BYTE (0
-255) Y%                                  PA
```

```
•9070 X%=3:Y%=150                              GF
•9071 POKE 54293,X%:POKE 54294,Y%              KJ
•9072 REM FILTER ON?                           DD
•9073 REM VOICE 1 ON=1; 2 ON=2; 3 ON=4; 1
 &2 ON=3; 2&3 ON=6; ALL ON=7                   AL
•9074 X%=0                                     FM
•9075 REM FILTER RESONANCE                     CE
•9076 REM PEAK VOLUME (0=LOW, 15=HIGH)         FH
•9077 Y%=14                                    JL
•9078 Y%=Y%*16:POKE 54295,X% OR Y%             MG
•9079 REM SELECT FILTER TYPE                   CP
•9080 REM LOW-PASS=1;BAND-PASS=2;HIGH-PAS
 S=4;LO-BAND=3;HI-BAND=6;ALL=7                 MH
•9081 X%=1                                     FP
•9082 X%=X%*16                                 LC
•9083 REM                                      JD
•9084 REM SELECT OVERALL VOLUME                FJ
•9085 REM 15=HIGH, 0=LOW                       PB
•9086 Y%=15                                    JK
•9087 POKE 54296,X% OR Y%                      HN
•9197 REM                                      JD
•9198 REM MACHINE LANGUAGE ROUTINE AT M        CE
•9199 REM                                      JD
•9218 REM FIND ADDRESS OF F$ AND PUT IT I
 N ZERO PAGE AT 139, 140                       MA
•9219 REM                                      JD
•9220 N=PEEK(45)+256*PEEK(46)+3:Y%=N/256:
 X%=N-Y%*256                                   IF
•9221 POKE M,173:POKE M+1,X%:POKE M+2,Y%:
 N=N+1:Y%=N/256:X%=N-Y%*256                    AN
•9222 POKE M+3,133:POKE M+4,251:POKE M+5,
 173:POKE M+6,X%:POKE M+7,Y%                   NK
•9223 POKE M+8,133:POKE M+9,252:POKE M+10
 ,160:POKE M+11,0                              HA
•9224 POKE M+12,162:POKE M+13,0                GG
•9228 REM                                      JD
•9229 REM GET EACH PITCH FROM F$ AND PUT
 IT IN FREQUENCY REGISTER                      CH
•9230 FOR I=M+14 TO M+54 STEP 8:POKE I,17
 7:POKE I+1,251                                KF
•9231 POKE I+2,157:POKE I+3,0:POKE I+4,21
 2                                             HK
•9232 POKE I+5,200:POKE I+6,162:READ A:PO
 KE I+7,A:NEXT                                 IE
•9233 DATA 1,7,8,14,15,4                       AK
•9238 REM                                      JD
•9239 REM GATE EACH SOUND OPEN                 MC
•9240 N=0:FOR I=M+59 TO M+71 STEP 6            BJ
•9241 POKE I,173:A=MM-3+N:Y%=A/256:X%=A-2
 56*Y%:POKE I+1,X%:POKE I+2,Y%                 PO
•9242 POKE I+3,141:READ A:POKE I+4,A:POKE
  I+5,212:N=N+1:NEXT                           DG
•9243 DATA 4,11,18                             PB
•9247 REM                                      JD
•9248 REM GARBAGE COLLECTION                   ME
•9250 FOR I=M+77 TO M+90:READ A:POKE I,A:
 NEXT                                          CE
•9255 DATA 164,52,165,51,105,6,144,1,200,
 133,51,132,52,96                              BN
•9257 REM                                      JD
•9258 REM GATE-OFF ML ROUTINE AT MM            LA
•9259 REM                                      JD
•9260 FOR I=MM TO MM+16 STEP 8                 GG
•9261 POKE I,173:Y%=3:X%=34+(I-MM)/8:POKE
 I+1,X%:POKE I+2,Y%:REM USES 820-822           BB
•9262 POKE I+3,41:POKE I+4,254                 DI
•9263 POKE I+5,141:READ A:POKE I+6,A:POKE
 I+7,212:NEXT:POKE MM+24,96                    KD
•9264 DATA 4,11,18                             PB
•9267 REM                                      JD
•9268 REM SET WAVEFORMS WITH GATES OFF         NK
•9269 REM STORED AT 820-822--LINE 9261 RE
 QUIRES THIS (820=HEX 03 34)                   BO
•9270 POKE 820,G%(0):POKE 821,G%(1):POKE
 822,G%(2):SYS MM                              OB
•9280 PH=0                                     HP
•9290 RETURN                                   IM
```

SUMMERTIME

```
•1 REM "SUMMERTIME"                            ON
•3 REM ADD THESE LINES TO "SONG MAKER" TO
 CREATE THE SONG "SUMMERTIME"                  CF
•5 REM FROM "PORGY AND BESS" BY GEORGE AN
 D IRA GERSHWIN                                LP
•7 REM FIRST DELETE ALL "SONG MAKER" LINE
 S ABOVE 9500 AND BELOW 9900                   EP
•9001 PH=0:ES%=3:DIM MD$(ES%),G%(2),AD(2)
 ,Y%(2)                                        OL
•9010 AK%(0)=2:AK%(1)=0:AK%(2)=4               LN
•9020 DY%(0)=5:DY%(1)=0:DY%(2)=8               LP
•9030 SN%(0)=4:SN%(1)=0:SN%(2)=1               LN
•9040 RE%(0)=2:RE%(1)=0:RE%(2)=3               CL
•9070 WF%(0)=16:WF%(1)=32:WF%(2)=64            EI
•9075 G%(0)=WF%(0)OR 7:G%(1)=WF%(1) OR 1:
 G%(2)=WF%(2) OR 1                             LC
•9080 PW%(0,0)=200:PW%(0,1)=12                 BP
•9081 PW%(1,0)=200:PW%(1,1)=12                 HL
•9082 PW%(2,0)=200:PW%(2,1)=3                  KF
•9105 DATA 24,128,192,256,384,512,640,768
 ,896,1024                                     PM
•9300 X%=3:Y%=150                              GF
•9310 X%=0                                     FM
•9320 Y%=14                                    JL
•9330 X%=1                                     FP
•9340 Y%=15:POKE 54296,X% OR Y%                EK
•9345 EV%=0                                    KC
•9499 REM PHRASE 0                             IP
•9500 MD$(0) = "0420093041[3"3"]5082"          BP
•9501 ME$(0,0)="[c E]ECE[c E]E@[s C]DCDEC
 A@E@"                                         IM
•9502 MV$(0,0)="4[12" "]3[3" "]"               OL
•9509 REM PHRASE 1                             IO
•9510 MD$(1) = "0430280[5"3"]9[6"0"]31"        AG
•9511 ME$(1,0)="FEC[s C]DD[c C]CACACBAGFE
 DCB@"                                         EB
•9512 MV$(1,0)="4[7" "]34343[6" "]2 "          JJ
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
·9519 REM PHRASE 2                              JB
·9520 MD$(2) = "04202406420[3"3"]5082"          GD
·9521 ME$(2,0)="[c E]EC[c E]EE[c E]EDC[s
 C]DECA[c E]E@"                                 IE
·9522 MV$(2,0)="4[13" "]3[3" "]"                DD
·9529 REM PHRASE 3                              JA
·9530 MD$(3) = "[4"4"]360[4"5"]09[4"0"]2"       FE
·9531 ME$(3,0)="EGEGAC[c E]EDCCBAF[s F]G[
 s G]A"                                         DA
·9532 MV$(3,0)="3[4" "]4[5" "]3 5[4" "]"        CG
```

VIC MUSIC

```
·1 REM "VIC MUSIC"                              CP
·10 MD$="@":M1$=MD$:M2$=MD$:M3$=MD$:GOSUB
 9000                                           PE
·100 SYS SR                                     EF
·190 END                                        IC
·9000 SR=828                                    PN
·9005 POKE 170,50 :REM TEMPO                    EG
·9006 POKE 659,1  :REM ATTACK                   BI
·9007 POKE 660,200:REM DECAY                    BO
·9008 POKE 661,15-0 :REM SUSTAIN VOLUME         IN
·9009 POKE 662,10 :REM RELEASE                  MF
·9010 FOR I=663 TO 676:READ A:POKE I,A:NE
 XT                                             PC
·9011 DATA 166,169,169,5,133,171,198,171,
 208,252,202,208,245,96                         EA
·9015 FOR I=677 TO 682:READ A:POKE I,A:NE
 XT                                             IG
·9016 DATA 202,234,234,208,251,96               NO
·9020 FOR I=684 TO 767:READ A:POKE I,A:NE
 XT                                             PC
·9025 DATA 32,151,2,136,208,250,96              FG
·9030 DATA 173,14,144,41,240,5,150,141,14
 ,144,96                                        CJ
·9035 DATA 160,15,230,150,32,179,2,174,14
 7,2,32,165,2,136,208,242                       PI
·9036 DATA 162,1 ,32,153,2                      BJ
·9037 DATA 172,149,2,198,150,32,179,2,174
 ,148,2,32,165,2,136,208,242                    AF
·9038 DATA 166,170,134,169,164,149,32,172
 ,2                                             JL
·9039 DATA 164,150,240,14,198,150,32,179,
 2,174,150,2,32,165,2,136,208,242,96            AP
·9045 FOR I=SR TO SR+89:READ A:POKE I,A:N
 EXT                                            LD
·9050 DATA 169,0,133,150,133,147,160,2,17
 7,45,133,148                                   EA
·9051 DATA 200,177,45,133,155,200,177,45,
 133,156                                        DE
·9052 DATA 160,10,177,45,133,158,200,177,
 45,133,159                                     CN
·9053 DATA 160,17,177,45,133,163,200,177,
 45,133,164                                     MG
·9054 DATA 160,24,177,45,133,167,200,177,
 45,133,168                                     GC
·9055 DATA 164,147,177,155,133,149              BC
·9056 DATA 177,158,141,10,144                   OA
·9057 DATA 177,163,141,11,144                   LP
·9058 DATA 177,167,141,12,144                   FC
·9059 DATA 32,190,2,169,0,133,150,230,147
 ,198,148,208,222,96                            DO
·9100 SL=48:REM # OF NOTES                      LF
·9110 FOR I=1 TO SL:READ B:B=B+128:IF B>2
 55 THEN B=0                                    PB
·9115 M3$=M3$+CHR$(B):NEXT                      MG
·9120 FOR I=1 TO SL:M1$=M1$+CHR$(0):NEXT        JH
·9125 FOR I=1 TO SL:MD$=MD$+CHR$(25):NEXT       FE
·9130 M2$=M1$                                   AE
·9200 DATA 87,81,79,67,67,47,67,67,79,67,
 79,87                                          CD
·9210 DATA 81,79,81,73,73,47,73,73,81,73,
 81,91                                          HH
·9220 DATA 87,81,79,67,67,47,67,67,79,67,
 79,87                                          CD
·9230 DATA 81,79,81,79,81,73,87,81,79,67,
 67,67                                          AC
·9990 RETURN                                    IM
```

F L A S H

FROM PAGE 77

This program requires use of an assembler. See introductory article for explanation.

```
*
* FLASH
*
              ORG     $8000
EOL           EQU     13
BUFLEN        EQU     40
FILLCH        EQU     $20
CHROUT        EQU     $FFD2
CLRSCR        EQU     147
*
              JMP     START
*
TEXT          ASC     'FLASH: C64 OWNER B
REAKS MACHINE CODE!'
              DFB     13
*
* CLEAR SCREEN
*
START         JSR     CLEAR
*
* CLEAR TEXT BUFFER
*
              LDA     #FILLCH
              LDX     #BUFLEN
STUFF         DEX
              STA     TXTBUF,X
              BNE     STUFF
*
```

```
* STORE MESSAGE IN BUFFER
*
            LDX    #0
LOOP1       LDA    TEXT,X
            STA    TXTBUF,X
            CMP    #EOL
            BEQ    PRINT
            INX
            CPX    #BUFLEN
            BCC    LOOP1
*
* PRINT MESSAGE
*
PRINT       LDX    #0
LOOP2       LDA    TXTBUF,X
            PHA
            JSR    CHROUT
            PLA
            CMP    #EOL
            BNE    NEXT
            JMP    AGAIN
NEXT        INX
            CPX    #BUFLEN
            BCC    LOOP2
*
AGAIN       JSR    DELAY
            JSR    CLEAR
            JSR    DELAY
            JMP    PRINT
*
* ROUTINE TO CLEAR SCREEN
*
CLEAR       LDA    #CLRSCR
            JSR    CHROUT
            RTS
*
* DELAY LOOP
*
DELAY       LDX    #$FF
XLOOP       LDY    #$FF
YLOOP       DEY
            NOP
            NOP
            NOP
            BNE    YLOOP
            DEX
            BNE    XLOOP
            RTS
*
TXTBUF      DS     BUFLEN
*
```

Etch

FROM PAGE 73

```
1 REM BEFORE LOADING POKE44,64:POKE256*6
4,0:NEW                                    FM
2 S0 = PEEK(46)*256+PEEK(45)               BF
3 S1 = S0-210:S2 = S0-150                  CF
9 PRINT"[CLEAR]""IN ADDITION TO NORMAL C
URSOR MOVEMENTS,"                          PC
10 PRINT"THE FOLLOWING CODES ARE USED: F
-FAST"                                     HF
11 PRINT"SCAN; S-SLOW SCAN; D-DRAW; E-ER
ASE"                                       PE
12 PRINT"I-INITIAL POINT OF SLANTED LINE
;"                                         HL
13 PRINT"L-LINE BACK TO INITIAL POINT; F
1-"                                        NJ
14 PRINT"DUMP SCREEN IMAGE TO PRINTER"     LK
15 PRINT"F7-CLEAR SCREEN"                  DH
16 INPUT "DO YOU WANT TO PLAY SQUARES";A
N$                                         KH
17 SYS S1                                  CF
18 X=0:Y=0:SV=0:BY=8192:F=1:D=0:E=0:GOSU
B 1000                                     BH
19 IF AN$="Y" THEN GOSUB 200               HJ
20 GET M$:IFM$="" THEN 20                  FJ
21 IF M$=CHR$(136) THEN 17                 KH
22 IF M$="F" THEN F=5:GOTO 20              AE
24 IF M$="S" THEN F=1:GOTO 20              HP
25 IF M$="D" THEN D=NOT D:E=0:GOTO 20      HD
26 IF M$="E" THEN E=NOT E:D=0:GOTO20       BM
27 IF M$="I" THEN XI=X:YI=Y:GOTO 20        AJ
28 IF M$="L" THEN 100                      IH
29 IF M$=CHR$(133) THEN 1100               PN
30 IF M$="[UP]" THEN Y=Y-F:IF Y<0 THEN Y
=0                                         LE
35 IF M$="[DOWN]" THEN Y=Y+F:IF Y>199 TH
EN Y=199                                   NG
40 IF M$="[RIGHT]" THEN X=X+F:IF X>319 T
HEN X=319                                  DC
45 IF M$="[LEFT]" THEN X=X-F:IF X<0 THEN
 X=0                                       KG
50 GOSUB 1000:GOTO 20                      HH
100 DX=X-XI : DY=Y-YI                      EM
110 M=DY/DX : B=YI-M*XI                    AM
115 SP = (SGN(XI-X))*F : XF=X              FA
120 FOR X=XF TO XI STEP SP                 LD
130 Y = INT(M*X+B)                         FG
140 GOSUB 1000                             FO
150 NEXT                                   IA
160 D=0:GOTO 20                            EG
200 D=1:FOR Y = 0 TO 199 STEP 21           HN
210 FOR X = 0 TO 249 STEP 25               DH
220 GOSUB 1000                             FO
230 NEXTX : NEXTY                          PL
240 X = 128 : Y = 95 : GOSUB 1000          BP
250 D=0 : RETURN                           PA
1000 CH=INT(X/8):RO=INT(Y/8):LN=YAND7      FK
1010 IF D=0 THEN  POKE BY,SV               FO
1015 IF E=0 THEN 1020                      CO
1017 Z = SV AND (2[UPARROW]BT):POKE BY,S
V-Z                                        EE
1020 BY=8192+RO*320+CH*8+LN:BT=7-(XAND7)   KH
1030 SV = PEEK(BY)                         BC
```

To enter *The Wizard of Im...* you must use our *Flankspeed* machine language entry program. See the introduction to *Flankspeed* on page 86 of this issue.

```
·1040 POKE BY,SV OR(2[UPARROW]BT)             LK
·1050 RETURN                                  IM
·1100 OPEN 4,4,5:CMD4                         KJ
·1110 PRINT CHR$(27)CHR$(51)CHR$(16)          EC
·1120 POKE 251,0:POKE 252,32                  KO
·1130 FOR I=0 TO 7:POKE 2048+I,2[UPARROW]
 I:NEXT                                       FP
·1140 FOR L=0 TO 24:PRINT CHR$(27)CHR$(76
 )CHR$(250)CHR$(2);                           IA
·1150 SYS S2                                  CE
·1160 PRINT CHR$(245)CHR$(10);                PC
·1170 NEXT L                                  MO
·1180 PRINT#4,CHR$(27)"@":PRINT#4,:CLOSE4     GL
·1190 END                                     IC
```

Machine code to be appended to ETCH
(see article for instructions)

```
169 24 141 24 208 169 32 13 17 208 141 17
208 169 32 133 252 160 0 132 251 169 0 145
251 200 208 251 166 252 232 134 252 224 64
208 242 169 4 133 252 132 251 169 1 145
251 200 208 251 166 252 232 134 252 224 8
208 242 96 162 0 169 0 32 210 255 232 224
120 208 246 169 245 32 210 255 169 0 141 8
8 141 9 8 141 10 8 141 11 8 169 7 56 237 9
8 170 189 0 8 141 12 8 169 7 56 237 10 8
170 189 0 8 141 13 8 172 10 8 177 251 45
12 8 240 10 173 13 8 24 109 11 8 141 11 8
174 10 8 232 142 10 8 224 8 208 212 173 11
8 32 210 255 32 210 255 169 0 141 10 8 141
11 8 174 9 8 232 142 9 8 224 8 208 171 169
0 141 9 8 169 8 24 101 251 133 251 144 5
166 252 232 134 252 174 8 8 232 142 8 8
224 40 208 141 96
```

THE WIZARD OF IM

FROM PAGE 47

First Byte: C000 Last Byte: C71F Sys to Start: 49152

```
C000: A9 0F 8D 18 D4 20 DF C2 F5
C008: 20 27 C3 A9 00 8D 22 C7 34
C010: 8D 23 C7 20 44 E5 A9 00 7C
C018: A0 17 99 00 D4 88 D0 FA 92
C020: A0 26 A9 0A 8D FE C6 A9 97
C028: 00 85 FC A9 D8 85 FD A2 53
C030: 16 AD FE C6 91 FC 18 A5 06
C038: FC 69 28 85 FC A5 FD 69 56
C040: 00 85 FD CA D0 EB CE FE 19
C048: C6 AD FE C6 C9 01 F0 0C 4A
C050: 88 88 D0 D3 8C 00 D8 84 EF
C058: FE 4C 64 C0 A9 0A 8D FE 09
C060: C6 4C 50 C0 A2 78 A9 0B 54
C068: 9D 6F DB A9 A0 9D 6F 07 AF
C070: CA D0 F3 18 A2 16 A0 0E 7F
C078: 20 F0 FF A0 00 B9 66 C6 11
C080: F0 07 20 D2 FF C8 4C 7D FD
C088: C0 A9 A6 8D 00 D0 A9 FD 9F
C090: 8D F8 07 A9 00 8D 20 C7 3D
C098: 8D 21 C7 8D FF C6 8D 10 01
C0A0: D0 8D 26 C7 AD 1F D0 A9 34
C0A8: 01 8D 25 C7 A9 03 8D 24 82
C0B0: C7 8D 1E C7 8D 1D C7 A9 08
C0B8: 05 8D 1C C7 A9 30 8D 2B C1
C0C0: C7 A9 06 8D 2D C7 8D 2C 74
C0C8: C7 A9 02 8D 2E C7 A9 20 89
C0D0: 8D 28 C7 20 B2 C4 20 E4 EA
C0D8: FF C9 8C F0 45 C9 88 F0 A8
C0E0: 6F AD 8D 02 C9 01 F0 EE 38
C0E8: AD 1F D0 29 01 D0 64 CE B4
C0F0: 29 C7 F0 4F CE 2A C7 F0 D3
C0F8: 54 AD 00 DC 49 FF 8D FD AC
C100: C6 29 10 C9 10 F0 36 CE CF
C108: 27 C7 D0 CA AD 28 C7 8D BD
C110: 27 C7 AD FD C6 29 0F C9 73
C118: 04 F0 67 C9 08 F0 37 4C BA
C120: D6 C0 A9 06 8D 21 D0 A9 90
C128: 0E 8D 20 D0 8D 86 02 A9 74
C130: 00 8D 15 D0 8D 18 D4 20 3E
C138: 44 E5 20 74 A4 4C B0 C1 5A
C140: 4C 82 C1 AD 15 D0 29 02 8F
C148: F0 AA 4C D1 C1 4C 0E C2 E0
C150: 4C 13 C0 4C C9 C4 A9 FD F2
C158: 8D F8 07 EE 00 D0 F0 0A A0
C160: AD 00 D0 C9 45 F0 0E 4C 39
C168: D6 C0 AD 10 D0 09 01 8D 26
C170: 10 D0 4C D6 C0 AD 10 D0 C3
C178: 29 01 F0 03 CE 00 D0 4C 82
C180: D6 C0 A9 FE 8D F8 07 CE 1D
C188: 00 D0 AD 00 D0 C9 FF F0 92
C190: 07 C9 18 F0 0E 4C D6 C0 5C
C198: AD 10 D0 29 02 8D 10 D0 C0
C1A0: 4C D6 C0 AD 10 D0 29 01 3D
C1A8: D0 03 EE 00 D0 4C D6 C0 20
C1B0: AD 15 D0 C9 03 F0 17 A9 C2
C1B8: 03 8D 15 D0 AD 00 D0 29 D6
C1C0: F8 8D 02 D0 AD 10 D0 0A B2
C1C8: 6D 10 D0 8D 10 D0 4C 07 D8
C1D0: C1 A9 12 8D 29 C7 CE 03 9E
C1D8: D0 AD 03 D0 C9 28 B0 15 E2
C1E0: A9 01 8D 15 D0 A9 BE 8D F4
C1E8: 03 D0 AD 10 D0 29 01 8D 03
C1F0: 10 D0 4C D6 C0 20 9C C3 36
C1F8: 4C F4 C0 20 97 E0 A5 8E C7
C200: 29 0F 8D FE C6 A5 8D 29 E7
C208: 03 6D FE C6 0A 60 AE FF 57
C210: C6 F0 22 AD 2B C7 8D 2A 42
C218: C7 A5 FE 85 FC 38 E9 28 51
C220: 85 FE A5 FF 85 FD E9 00 B7
C228: 85 FF A0 26 20 8F C3 CE B6
C230: FF C6 4C F9 C0 CE 2C C7 C0
C238: D0 0F AD 2D C7 8D 2C C7 3C
C240: 20 FB C1 A8 A9 51 99 00 5B
```

```
C248: 04 A9 48 85 FE A9 07 85 F8
C250: FF A0 26 B1 FE C9 20 F0 A2
C258: 1E A9 20 91 FE AD 21 C7 67
C260: F0 15 38 AD 20 C7 ED 26 48
C268: C7 8D 20 C7 AD 21 C7 E9 26
C270: 00 8D 21 C7 4C 7B C2 88 F9
C278: 88 D0 D8 A2 15 8E FF C6 B7
C280: 4C F9 C0 38 AD 03 D0 E9 2B
C288: 32 4A 4A 4A 8D 30 C7 0A 29
C290: 0A 6D 30 C7 A2 00 8E 30 61
C298: C7 0A 0A 2E 30 C7 0A 2E D2
C2A0: 30 C7 85 59 AD 30 C7 69 86
C2A8: 04 85 5A AD 10 D0 4A 8D F2
C2B0: 32 C7 38 AD 02 D0 E9 14 61
C2B8: 8D 31 C7 AD 32 C7 E9 00 D0
C2C0: 4A 6E 31 C7 AD 31 C7 4A 63
C2C8: 4A 65 59 85 59 A5 5A 69 1A
C2D0: 00 85 5A 60 A9 FF 8D 00 48
C2D8: C7 CE 00 C7 D0 FB 60 A0 05
C2E0: FF C8 B9 66 C5 99 00 3F 68
C2E8: C0 FF D0 F5 A9 A6 8D 00 4E
C2F0: D0 A9 CD 8D 01 D0 A9 03 45
C2F8: 8D 1C D0 A9 06 8D 27 D0 A8
C300: A9 08 8D 25 D0 A9 01 8D 6D
C308: 15 D0 8D 26 D0 A9 FD 8D A7
C310: F8 07 A9 FF 8D F9 07 A9 F1
C318: 64 8D 02 D0 A9 C0 8D 03 D7
C320: D0 A9 07 8D 28 D0 60 20 A8
C328: 44 E5 A9 17 8D 18 D0 A9 33
C330: 00 8D 20 D0 8D 21 D0 A2 D0
C338: 06 A0 0B 18 20 F0 FF A0 B3
C340: 00 B9 C9 C6 F0 07 20 D2 75
C348: FF C8 4C 41 C3 A2 08 A0 AD
C350: 0C 18 20 F0 FF A0 00 B9 DF
C358: DB C6 F0 07 20 D2 FF C8 AE
C360: 4C 57 C3 A2 0A A0 0A 18 37
C368: 20 F0 FF A0 00 B9 EA C6 85
C370: F0 07 20 D2 FF C8 4C 6D DD
C378: C3 A2 05 A0 FF 20 D4 C2 3C
C380: 88 D0 FA CA D0 F5 20 44 CA
C388: E5 A9 15 8D 18 D0 60 B1 B5
C390: FE 91 FC A9 20 91 FE 88 01
C398: 88 D0 F4 60 20 83 C2 A2 50
C3A0: 00 A1 59 C9 20 F0 07 A9 27
C3A8: 20 81 59 4C BA C3 A0 01 10
C3B0: B1 59 C9 20 F0 3C A9 20 9C
C3B8: 91 59 A9 FC 8D F9 07 A9 82
C3C0: 20 81 59 20 F3 C3 A9 05 42
C3C8: 8D 2F C7 20 D4 C2 CE 2F 03
C3D0: C7 D0 F8 A9 80 8D 04 D4 F2
C3D8: A9 01 8D 15 D0 AD 10 D0 85
C3E0: 29 01 8D 10 D0 A9 FF 8D B0
C3E8: F9 07 A9 C0 8D 03 D0 20 D5
C3F0: 0D C4 60 A9 3C 8D 00 D4 6B
C3F8: A9 C8 8D 01 D4 A9 81 8D 87
C400: 04 D4 A9 8E 8D 05 D4 A9 22
C408: D5 8D 06 D4 60 18 AD 20 8C
C410: C7 6D 1C C7 8D 20 C7 AD 4C
C418: 21 C7 69 00 8D 21 C7 20 01
C420: 6F C4 20 26 C4 60 AD 21 8E
C428: C7 CD 1D C7 D0 40 AC 25 85
C430: C7 B9 03 C7 8D 1C C7 18 06
C438: AD 1D C7 6D 1E C7 8D 1D C8
C440: C7 EE 1E C7 EE 26 C7 EE A8
C448: 25 C7 20 9C C4 CE 2B C7 78
C450: CE 2B C7 B0 05 A9 01 8D FF
C458: 2B C7 CE 2E C7 D0 0F CE BE
C460: 2D C7 D0 05 A9 01 8D 2D 90
C468: C7 A9 04 8D 2E C7 60 A2 64
C470: 18 A0 04 18 20 F0 FF A9 FF
C478: 0E 8D 86 02 AD 21 C7 AE E1
C480: 20 C7 20 CD BD 60 A2 18 2F
C488: A0 0F 18 20 F0 FF A9 07 12
C490: 8D 86 02 AD 24 C7 69 30 D9
C498: 20 D2 FF 60 A2 18 A0 18 5F
C4A0: 18 20 F0 FF A9 05 8D 86 8C
C4A8: 02 A9 00 AE 25 C7 20 CD DD
C4B0: BD 60 A2 18 A0 1F 18 20 81
C4B8: F0 FF A9 08 8D 86 02 AD 1F
C4C0: 23 C7 AE 22 C7 20 CD BD EF
C4C8: 60 20 0E C5 AD F8 07 8D 58
C4D0: 1F C7 A9 01 8D 15 D0 20 F5
C4D8: 4C C5 A9 32 8D 2F C7 A9 F4
C4E0: FC 8D F8 07 20 D4 C2 20 43
C4E8: D4 C2 AD 1F C7 8D F8 07 A2
C4F0: 20 D4 C2 CE 2F C7 D0 E7 27
C4F8: A9 80 8D 04 D4 CE 24 C7 44
C500: 20 86 C4 AD 1F D0 AE 24 DB
C508: C7 F0 0E 4C D6 C0 A9 20 7C
C510: A0 78 99 F7 06 88 D0 FA 15
C518: 60 20 6F C4 AD 21 C7 CD 31
C520: 23 C7 90 19 D0 08 AD 20 5B
C528: C7 CD 22 C7 90 0F AD 20 15
C530: C7 8D 22 C7 AD 21 C7 8D 93
C538: 23 C7 20 B2 C4 20 E4 FF BF
C540: F0 FB C9 8C F0 03 4C 13 D6
C548: C0 4C 22 C1 A9 5F 8D 00 CF
C550: D4 A9 07 8D 01 D4 A9 81 64
C558: 8D 04 D4 A9 3F 8D 05 D4 0F
C560: A9 08 8D 06 D4 60 00 30 0B
C568: 00 00 00 00 0C 10 C0 00 45
C570: 00 00 C2 22 0C 00 10 00 71
C578: 04 20 40 82 AA 08 00 B8 CA
C580: 00 2A FE A3 00 B8 00 82 88
C588: AA 08 04 20 40 00 10 00 AF
C590: C2 22 0C 00 00 00 0C 20 AD
C598: C0 00 00 00 00 10 00 00 69
C5A0: 00 00 00 30 00 00 A0 0C 7D
C5A8: 00 2A 0F 00 2A 83 C0 0A 5A
C5B0: 90 D0 0A 94 54 02 D0 14 EB
C5B8: 02 FC 04 02 80 04 0A AA F6
C5C0: 94 2A AA 94 2A AA 80 2A 3E
C5C8: A2 80 2A A0 80 16 A0 80 6E
C5D0: 1A A0 00 0A A0 00 0A A0 E0
C5D8: 00 0A A0 00 0A A0 00 0A 38
C5E0: A0 00 2A A8 00 10 00 30 94
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering *Ahoy!* programs. Refer to these pages **before** entering any programs!

```
C5E8: 2A 00 F0 A8 03 C1 A8 0F 29
C5F0: 05 A0 15 17 A0 14 07 80 FE
C5F8: 10 3F 80 10 0A 80 16 AA 24
C600: A0 16 AA A8 02 AA A8 02 61
C608: 8A A8 02 0A A8 02 0A 94 90
C610: 00 0A A4 00 0A A0 00 0A 73
C618: A0 00 0A A0 00 0A A0 00 0E
C620: 0A A0 00 2A A8 10 00 14 C1
C628: 00 00 55 00 01 69 40 01 29
C630: 7E 40 01 BE 40 01 BD 40 ED
C638: 01 6E 40 01 BE 00 00 99 41
C640: 00 00 AE 00 00 24 00 00 13
C648: 18 00 00 2C 00 00 24 00 B0
C650: 00 38 00 00 1C 00 00 18 BC
C658: 00 00 1C 00 00 30 00 00 A4
C660: 00 00 00 0C 00 00 12 97 16
C668: 57 49 5A 41 52 44 20 4F AA
C670: 46 20 49 4D 0D 12 20 20 CC
C678: 20 20 9F 53 43 4F 52 45 D5
C680: 20 20 20 20 4C 49 56 45 32
C688: 53 20 20 20 20 4C 45 56 44
C690: 45 4C 20 20 20 20 48 49 34
C698: 47 48 20 0D 9A 12 1D 1D 3C
C6A0: 1D 1D 30 20 20 20 20 98 24
C6A8: 20 20 20 20 9E 20 20 33 3B
C6B0: 20 20 98 20 20 20 20 1E 28
C6B8: 20 20 31 20 20 98 20 20 43
C6C0: 20 20 81 20 20 20 20 20 23
C6C8: 00 9E D4 C8 C5 20 D7 C9 8C
C6D0: DA C1 D2 C4 20 CF C6 20 DB
C6D8: C9 CD 00 42 59 20 20 C2 0F
C6E0: 4F 42 20 D3 50 49 52 4B 9D
C6E8: 4F 00 C3 4F 50 59 52 49 90
C6F0: 47 48 54 20 28 C3 29 20 2A
C6F8: 31 39 38 35 00 00 00 00 D0
C700: 00 00 00 00 08 0C 13 1E 45
C708: 30 4C 79 C1 1E 30 4C 79 D3
C710: C1 2D 48 73 B8 50 69 99 C6
C718: 64 78 C8 32 00 00 00 00 EF
```

Auto-Append

FROM PAGE 62

Bug Repellent line codes listed to right of program lines are for C-64 only! VIC 20 *Bug Repellent* line codes are listed at end of program.

```
•1 REM*****************               LO
•2 REM*                  *            OO
•3 REM*   AUTO-APPEND    *            PN
•4 REM*                  *            OO
•5 REM*        BY        *            IA
•6 REM*  WILLIAM BRAUN   *            GJ
•7 REM*       1984       *            MK
•8 REM*****************               LO
•10 POKE55,225:POKE56,PEEK(56)-1:A=PEEK(5
 6)*256+226:FORX=0TO27:READD:POKEA+X,D:NE
 XT                                   PG
•15 PRINT"[CLEAR][BLACK][DOWN][5" "]INSTR
 UCTIONS[BLUE][DOWN][DOWN]":PRINT" 1. LOA
 D 1ST PROGRAM":PRINT"[DOWN] 2. SYS"A  CA
•20 PRINT"[DOWN] 3. LOAD 2ND PROGRAM":PRI
 NT"[DOWN] 4. SYS"A+19:PRINT"[DOWN] 5. SA
 VE AND VERIFY                         AJ
•25 DATA165,43,133,251,165,44,133,252,166
 ,45,202,202,134,43,165,46,133,44,96   CM
•30 DATA165,251,133,43,165,252,133,44,96 KI
```

VIC 20 BUG REPELLENT LINE CODES

#1	:LO	#4	:OO	#7	:MK	#15	:LE	#30	:KI
#2	:OO	#5	:JO	#8	:LO	#20	:BI		
#3	:CC	#6	:HI	#10	:PG	#25	:CM		

AUTOS: LEASING V. BUYING

FROM PAGE 75

```
•10 PRINTCHR$(147),CHR$(144):POKE53281,14 NA
•20 PRINTTAB(9)"AUTO: LEASING VS BUYING"  OA
•25 PRINTTAB(9):FOR J=1TO23:PRINT CHR$(18
 3);:NEXT:PRINT:PRINT                  MM
•30 PRINTTAB(13)"COST OF BUYING":PRINT:PR
 INT                                   NH
•35 PRINTTAB(3)"PLEASE ENTER DATA AS REQU
 ESTED":PRINT:PRINT                    JP
•40 INPUT" ENTER THE PURCHASE PRICE $ ";P
 P:PRINT                               ND
•50 INPUT" WILL YOU NEED FINANCING? Y/N";
 F$:PRINT:IF F$="N" THEN 190           ON
•60 PRINT" DOWNPAYMENT INCLUDING TRADE-IN
 "                                     OL
•70 INPUT "   ALLOWANCE $ ";DP:PRINT :LA=P
 P-DP                                  HK
•80 PRINT"[3" "]YOU WILL NEED TO BORROW $
 ";LA:PRINT                            HG
•90 PRINT" WILL YOU NEED TO BORROW MORE T
 O"                                    DE
•95 PRINT"  COVER SALES TAX, LICENSE, ETC
 .?"                                   JC
•100 INPUT"[4" "]Y/N";H$:PRINT         LG
•105 IF H$="N" THEN LA=PP-DP: GOTO 140 EC
•110 PRINT"[3" "]ENTER ADDITIONAL AMOUNT
 BORROWED"                             IE
•120 INPUT "$ ";AB: LA=LA+AB :PRINT    MN
•140 PRINT" ANNUAL PERCENTAGE INTEREST RA
 TE?"                                  BG
•150 INPUT "[5" "]% ";R :PRINT         CO
•160 IF R>1 THEN R=R/100               BD
•170 PRINT" LENGTH OF LOAN IN MONTHS." LD
•180 INPUT "[5" "]MONTHS";M:PRINT      NK
•185 IF M>60 THEN PRINT"SORRY,60 MONTH LI
 MIT, PLEASE RE-ENTER":PRINT:GOTO 170  GP
•190 INPUT" SALES TAX RATE % ";SX:PRINT BB
•200 IF SX>1 THEN SX=SX/100            CA
•210 INPUT" ANNUAL LICENSE FEE $ ";LC:PRI
```

```
NT                                         MD
·220 PRINT" WHAT TYPE OF CAR ARE YOU BUYI
NG?":PRINT                                 FO
·230 PRINT"[3" "]1. STANDARD"              FF
·240 PRINT"[3" "]2. COMPACT"               MN
·250 PRINT"[3" "]3. SUBCOMPACT?":PRINT     PJ
·260 INPUT"PLEASE ENTER NUMBER";N:PRINT    KO
·270 PRINT" DO YOU ITEMIZE YOUR DEDUCTION
S FOR"                                     PO
·280 INPUT"  INCOME TAX PURPOSES? Y/N";Y$
:PRINT                                     LN
·290 IF Y$="N" THEN 320                    IJ
·300 PRINT" ENTER YOUR TAX BRACKET RATE"   HF
·310 INPUT"%";TX:PRINT                     KM
·320 IF TX>1 THEN TX=TX/100                AN
·370 PRINTCHR$(147)                        FG
·380 PRINTTAB(12)"COST OF LEASING"         DB
·390 PRINTTAB(12):FORK=1TO15:PRINTCHR$(18
3);:NEXT:PRINT:PRINT                       NO
·400 PRINTTAB(3)"PLEASE ENTER DATA AS REQ
UESTED":PRINT:PRINT                        JP
·410 PRINT" ENTER SECURITY DEPOSIT REQUIR
ED"                                        GF
·415 INPUT"[5" "]$";SD:PRINT               BN
·420 PRINT" ENTER MONTHLY LEASE PAYMENT"   HD
·425 INPUT"[5" "]$";ML:PRINT               ED
·430 PRINT" FOR HOW MANY MONTHS?"          DG
·435 INPUT"[5" "]MONTHS";NP:PRINT:IF F$="
N" THEN M=NP                               CH
·440 PRINT" DOWNPAYMENT OR ADDITIONAL PAY
MENT"                                      ON
·450 INPUT"[5" "]$";AP:PRINT               BL
·460 INPUT" ANNUAL LICENSE FEE $";LF:PRIN
T                                          JH
·470 GOSUB 1000                            FO
·480 GOSUB 1300                            FB
·510 GOSUB 1500:PRINT:PRINT                CH
·520 INPUT"PRESS 'C' TO CONTINUE";C$       GP
·530 GOSUB 1800:PRINT:PRINT                PO
·540 INPUT"PRESS 'C' TO CONTINUE";C$       GP
·550 PRINTCHR$(147):POKE 53280,2           IN
·560 PRINTTAB(5)"COMPARISON: LEASING VS.
BUYING"                                    DL
·570 PRINTTAB(5):FORK=1TO30:PRINTCHR$(183
);:NEXT:PRINT:PRINT                        ND
·590 MP=INT(MP*100+.5)/100:X$=STR$(MP):GO
SUB 3000                                   GC
·600 PRINT"YOUR MONTHLY LOAN PAYMENTS WOU
LD BE:":PRINT                              HK
·610 PRINTTAB(10)"$";X$:PRINT              LC
·620 PRINT"YOUR MONTHLY LEASE PAYMENT WOU
LD BE:":PRINT                              AD
·630 PRINTTAB(10)"$";ML:PRINT:PRINT        ID
·640 X$=STR$(TT):GOSUB 3000                IJ
·650 PRINT"THE NET COST OF BUYING IS  $";
SPC(SP)X$:PRINT                            AF
·660 X$=STR$(LT):GOSUB 3000                IB
·670 PRINT"THE NET COST OF LEASING IS $";
SPC(SP)X$                                  NN
·675 PRINTTAB(28);:FOR K=1TO9:PRINTCHR$(1
83);:NEXT:PRINT:PRINT                      AD
·680 DF=LT-TT:X$=STR$(DF):GOSUB 3000       FD
·690 IF DF>0 THEN PRINT"[5" "]BUYING IS C
HEAPER BY  $";SPC(SP)X$                    OB
·700 IF DF<0 THEN PRINT"[5" "]LEASING IS
CHEAPER BY $";SPC(SP)VAL(X$)*-1            PG
·900 END                                   IC
·1000 REM COST OF BUYING CALCULATIONS      JN
·1010 YR=M/12:MR=R/12:T=INT(YR) :TT=0:TD=
0                                          GG
·1015 XL=(PP*(1+.055)[UPARROW]YR)-PP       BF
·1020 LI=(DP*(1+.055)[UPARROW]YR)-DP       HA
·1025 IF LA=0 THEN 1070                    HP
·1030 F=(MR*(1+MR)[UPARROW]M)/((1+MR)[UPA
RROW]M-1)                                  NB
·1040 MP=F*LA                              NH
·1050 IX=(MP*M)-LA                         JF
·1060 IS=IX*TX                             EI
·1070 SS=(SX*PP)*TX                        AC
·1075 IF T>5 THEN PRINT"SORRY, 60 MONTH L
IMIT":END                                  KJ
·1080 IF N=1 THEN GOSUB 2000               MI
·1090 IF N=2 THEN GOSUB 2100               LK
·1100 IF N=3 THEN GOSUB 2200               LM
·1110 LC=LC*YR                             DC
·1120 RETURN                               IM
·1300 REM COMPUTE COST OF LEASING          LP
·1310 TL=ML*NP                             DA
·1320 T1=INT(NP/12)                        AC
·1330 IL=(SD*(1+.055)[UPARROW]T1)-SD       BM
·1340 IA=(AP*(1+.055)[UPARROW]T1)-AP       DH
·1350 LF=LF*T1                             PK
·1360 RETURN                               IM
·1500 REM DISPLAY BUYING RESULTS           JF
·1505 PRINTCHR$(147)                       FG
·1510 PRINTTAB(13)"COST OF BUYING"         BD
·1515 PRINTTAB(13);:FOR K=1TO14:PRINTCHR$
(183);:NEXT:PRINT:PRINT                    HM
·1520 IF F$="N" THEN 2500                  LE
·1525 DP=INT(DP):X$=STR$(DP):GOSUB 3000:T
T=TT+DP                                    LO
·1530 PRINT"DOWNPAYMENT[18"."]$";SPC(SP)X
$                                          IF
·1540 LI=INT(LI):X$=STR$(LI):GOSUB 3000:T
T=TT+LI                                    HO
·1550 PRINT"INTEREST LOST ON DOWNPAYMENT.
.";SPC(SP)X$                               ED
·1560 TP=(M*MP)-AB:TP=INT(TP):X$=STR$(TP)
:GOSUB 3000 :TT=TT+TP                      AI
·1570 PRINT"TOTAL PAYMENTS ON LOAN[8"."]"
;SPC(SP)X$                                 HK
·1575 PRINT"[SS][SS](AUTO PORTION ONLY)    EH
·1580 SA=SX*PP:SA=INT(SA):X$=STR$(SA):GOS
UB 3000: TT=TT+SA                          PH
·1590 PRINT"SALES TAX ON PURCHASE[9"."]";
SPC(SP)X$                                  MP
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering *Ahoy!* programs. Refer to these pages **before** entering any programs!

```
•1593 LC=INT(LC):X$=STR$(LC):GOSUB 3000:T
T=TT+LC                                         CE
•1595 PRINT"LICENSE FEES FOR ";INT(YR);"Y
EARS[5"."]";SPC(SP)X$                           PD
•1600 PRINTTAB(29);:FOR K=1TO10:PRINT CHR
$(183);:NEXT                                    NC
•1610 X$=STR$(TT):GOSUB 3000                    LD
•1620 PRINT:PRINTTAB(5)"TOTAL";SPC(19)"$"
;SPC(SP);X$                                     GN
•1630 PRINT:PRINT"LESS:":PRINT                  OL
•1640 Z=PP*(1-TD):Z=INT(Z):X$=STR$(Z):GOS
UB 3000:TT=TT-Z                                 GO
•1650 PRINT"ESTIMATED RESIDUAL VALUE"           HH
•1660 PRINT"[5" "]OF THE CAR[14"."]$";SPC
(SP)X$                                          KM
•1665 IF Y$="N" THEN 1720                       BF
•1670 IS=INT(IS):X$=STR$(IS):GOSUB 3000:T
T=TT-IS                                         AB
•1680 PRINT"TAX SAVINGS ON INTEREST[7"."]
";SPC(SP)X$                                     OF
•1700 SS=INT(SS):X$=STR$(SS):GOSUB 3000:T
T=TT-SS                                         ON
•1710 PRINT"TAX SAVINGS ON SALES TAX[6"."
]";SPC(SP)X$                                    HF
•1720 PRINTTAB(29):FORK=1TO10:PRINTCHR$(1
83);:NEXT:PRINT                                 EN
•1730 X$=STR$(TT):GOSUB 3000                    LD
•1740 PRINTTAB(5)"NET COST OF BUYING";SPC
(6)"$";SPC(SP)X$                                OJ
•1750 RETURN                                    IM
•1800 REM DISPLAY LEASING RESULTS               JP
•1810 PRINTCHR$(147)                            FG
•1820 PRINTTAB(12)"COST OF LEASING"             LO
•1830 PRINTTAB(12):FORK=1TO15:PRINTCHR$(1
83);:NEXT:PRINT:PRINT                           NO
•1840 SD=INT(SD):X$=STR$(SD):GOSUB 3000:L
T=LT+SD                                         BI
•1850 PRINT"SECURITY DEPOSIT[13"."]$"SPC(
SP)X$                                           IB
•1860 TL=INT(TL):X$=STR$(TL):GOSUB 3000:L
T=LT+TL                                         EI
•1870 PRINT"TOTAL OF LEASE PAYMENTS[7"."]
";SPC(SP)X$                                     DG
•1880 IL=INT(IL):X$=STR$(IL):GOSUB 3000:L
T=LT+IL                                         GM
•1890 PRINT"INTEREST LOST ON SECURITY DEP
.";SPC(SP)X$                                    ON
•1891 IF AP=0 THEN1906                          MK
•1895 AP=INT(AP):X$=STR$(AP):GOSUB 3000:L
T=LT+AP                                         CE
•1898 PRINT"ADDITIONAL REQUIRED PAYMENT[3
"."]";SPC(SP)X$                                 MK
•1900 IA=INT(IA):X$=STR$(IA):GOSUB 3000:L
T=LT+IA                                         ME
•1903 PRINT"LOST INTEREST ON ADD'L PAYMEN
T";SPC(SP)X$                                    EG
•1906 LF=INT(LF):X$=STR$(LF):GOSUB 3000:L
T=LT+LF                                         DE
•1907 PRINT"LICENSE FEE FOR ";T1;"YEARS[6
"."]";SPC(SP)X$                                 EM
•1909 PRINTTAB(29):FORK=1TO10:PRINTCHR$(1
83);:NEXT:PRINT                                 EN
•1910 LT=INT(LT):X$=STR$(LT):GOSUB 3000         BB
•1920 PRINT TAB(5)"TOTAL";SPC(19)"$";SPC(
SP)X$                                           IH
•1930 PRINT:PRINT"LESS:":PRINT                  OL
•1940 X$=STR$(SD):GOSUB 3000:LT=LT-SD           GK
•1950 PRINT"RETURN OF SECURITY DEPOSIT[3"
."]$";SPC(SP)X$                                 NN
•1955 PRINTTAB(29):FORK=1TO10:PRINTCHR$(1
83);:NEXT:PRINT                                 EN
•1960 LT=INT(LT):X$=STR$(LT):GOSUB 3000         BB
•1970 PRINTTAB(5)"NET COST OF LEASING";SP
C(5)"$";SPC(SP)X$                               NF
•1990 RETURN                                    IM
•2000 A(1)=.279:A(2)=.146:A(3)=.111:A(4)=
.099:A(5)=.084                                  OD
•2010 FOR K=1TOT:TD=TD+A(K):NEXT:RETURN         DF
•2100 B(1)=.167:B(2)=.126:B(3)=.106:B(4)=
.099:B(5)=.095                                  MA
•2110 FOR K=1TOT:TD=TD+B(K):NEXT:RETURN         EI
•2200 C(1)=.123:C(2)=.115: C(3)=.114:C(4)
=.107:C(5)=.106                                 GM
•2220 FOR K=1TOT:TD=TD+C(K):NEXT:RETURN         NL
•2500 REM DISPLAY CASH PURCHASE RESULTS         IE
•2510 PP=INT(PP):X$=STR$(PP):GOSUB 3000:T
T=TT+PP                                         AI
•2520 PRINT"PURCHASE PRICE[15"."]$";SPC(S
P)X$                                            MH
•2530 XL=INT(XL):X$=STR$(XL):GOSUB 3000:T
T=TT+XL                                         PA
•2540 PRINT"INTEREST LOST ON PUR. PRICE[3
"."]";SPC(SP)X$                                 AC
•2550 GOTO 1580                                 HG
•3000 SP=9-LEN(X$):RETURN                       KA
```

Script Analysis

FROM PAGE 17

```
•10 POKE53280,1:POKE53281,1:PRINT"[CLEAR]
[c 4]":DIMT1(20),T2(30)                         AK
•20 DIMW$(74),G$(14),G(14),W(14,7,9),T(14
,9),T$(14,9),P$(14),P(74)                       MN
•25 D$=".":N$=CHR$(13):CO$=", "                 NM
•30 FORI=0TO22:K$=K$+"[DOWN]":NEXT:K$="[H
OME]"+K$+"[RED][12"[RIGHT]"]PRESS ANY KE
Y"                                              KL
•87 S=2:C=12:O=(49152+2*1024)/256:PC=6143
9                                               KD
•88 FORI=1TO33:READA:POKE49151+I,A:NEXT:G
OSUB95                                          FE
•89 POKE56576,PEEK(56576)AND252:POKE53272
```

```
,S*16+C:POKE648,0                              LM
•90 FORI=53236TO53245:READA:POKEI,A:NEXT       DE
•91 POKE53246,PEEK(792):POKE53247,PEEK(79
 3):POKE792,244:POKE793,207:GOTO100            HG
•95 POKE56334,PEEK(56334)AND254:POKE1,PEE
 K(1)AND251:SYS49152                           NE
•96 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334
 )OR1:RETURN                                   FN
•97 DATA169,0,133,251,133,253,169,208,133
 ,252,169,240,133,254,162,16                   DL
•98 DATA160,0,177,251,145,253,136,208,249
 ,230,252,230,254,202,208,240,96               AJ
•99 DATA72,169,4,141,136,02,104,108,254,2
 07                                            HL
•100 FORI=1TO74:READW$(I):NEXT                 HH
•110 PRINT"[CLEAR][c 4]"TAB(17)"[7"[DOWN]
 "][RVSON]SCRIPT":PRINTTAB(16)"[DOWN][RVS
 ON]ANALYSIS                                   IG
•114 PRINTTAB(13)"[4"[DOWN]"][RVSON]BY BO
 B SPIRKO"                                     EH
•120 FORI=0TO14:READG$(I),G(I):NEXT            GG
•140 E=8:FORI=1TO49:READL:FORJ=1TO8:READK
 :POKEPC+L*E+J,K:NEXT:NEXT                     BN
•150 PRINT"[HOME]"TAB(19)"[8"[DOWN]"][BAC
 KARROW] [EP]":PRINTTAB(20)"[DOWN]]"           GC
•155 PRINTTAB(13)"[DOWN][DOWN][UPARROW]
 [UPARROW] [UPARROW][5" "][UPARROW]":PRIN
 TTAB(14)"[DOWN]][6" "][EP]"                   LF
•160 FORI=1TO53:READL:FORJ=1TO8:READK:POK
 EPC+L*E+J,K:NEXT:NEXT                         PJ
•170 FORI=0TO14:FORJ=0TOG(I):READT$(I,J):
 NEXT:NEXT                                     OC
•180 FORI=0TO14:FORJ=0TOG(I):READT(I,J):F
 ORK=0TOT(I,J):READW(I,J,K)                    BO
•190 NEXT:NEXT:NEXT                            PI
•200 INPUT"[CLEAR][c 5][DOWN][DOWN]FILENA
 ME";F$:PRINT"[DOWN][c 1]N[c 5]EW OR [c 1
 ]L[c 5]OAD?"                                  DN
•220 GETA$:IFA$=""THEN220                      HL
•230 IFA$="L"THEN4000                          LO
•240 PRINT"[DOWN][c 1]L[c 5]EFT- OR [c 1]
 R[c 5]IGHTHANDED?"                            KE
•242 GETH$:IFH$=""THEN242                      HN
•245 IFH$="L"THEN500                           GI
•250 H$="R":GOTO500                            CP
•300 B=VAL(A$)-1:FORC=0TOT(A,B):P(W(A,B,C
 ))=P(W(A,B,C))+1:NEXT:RETURN                  NF
•333 GETB$:IFB$=""THEN333                      FI
•337 IFB$<>"1"THENRETURN                       BJ
•339 P$=P$+T$(A,B)+CO$:A$=CHR$(B+49):GOSU
 B300:RETURN                                   FJ
•344 PRINTTAB(5)T$(A,B)":  [RVSON]";:RETU
 RN                                            JM
•400 PRINT"[c 4]":FORB=0TOG(A):PRINTTAB(7
 )RIGHT$(STR$(B+1),1)". "T$(A,B):NEXT          LL
•405 PRINT"[c 7]"TAB(7);                       KM
•410 PRINT"[RVSON][c B][LEFT]";:GETA$:PRI
 NT"[RVSOFF][c B][LEFT]";:IFA$=""THEN410       PB
•420 V=ASC(A$)-49:IFV<0ORV>G(A)THEN410         AM
•430 PRINT" ";:RETURN                          OA
•450 PRINTLEFT$(P$(I),J):PRINTRIGHT$(P$(I
 ),LEN(P$(I))-J):RETURN                        HG
•500 FORA=0TO14:AA=A+1:PRINT"[CLEAR][BLUE
 ][DOWN]"AA;G$(A)"[DOWN][c 5]"                 CD
•510 ONAAGOSUB1000,1200,1400,1600,1800,20
 00,2200,2400,2600,2800,3000,3200,3400         BF
•520 IFA=13THENGOSUB3600                       NE
•525 IFA=14THENGOSUB3800                       PD
•530 NEXT                                      IA
•600 PRINT"[CLEAR][DOWN][DOWN][BLUE]"F$":
  SCRIPT CHARACTERISTICS[DOWN][c 5]":FORI
 =0TO14                                        ON
•610 PRINT"[c 5]"G$(I)": [c 4]";:IFI=4THE
 N650                                          BP
•620 IFLEN(P$(I))<24THEN650                    BJ
•630 FORJ=20TO40:IFMID$(P$(I),J,1)=","THE
 NGOSUB450:GOTO660                             JD
•640 NEXTJ                                     MM
•650 PRINTP$(I)                                KN
•660 NEXTI:IFA$="T"ORA$="D"THENPRINTK$:GO
 TO745                                         DC
•700 M=0:N=0:FORI=1TO74:IFP(I)=.THEN740        FH
•720 IFP(I)>2THENM=M+1:T1(M)=I:GOTO740         PC
•730 IFP(I)=2THENN=N+1:T2(N)=I                 LN
•740 NEXT:PRINTK$                              ME
•745 GETA$:IFA$=""THEN745                      KF
•746 PRINT"[CLEAR][BLUE]"F$": ANALYSIS"        OK
•750 Y=44:PRINT"[DOWN][DOWN][c 4]DOMINANT
  TRAITS:[c 5]";                               GC
•760 FORI=1TOM:X$=W$(T1(I)):Y=Y-2*(I=M):G
 OSUB950                                       EN
•770 NEXT:Y=44:PRINT:PRINT"[c 4][DOWN]OTH
 ER TRAITS:[c 5]";                             NH
•780 FORI=1TON:X$=W$(T2(I)):Y=Y-2*(I=N):G
 OSUB950:NEXT                                  OB
•800 PRINT:PRINT"[4"[DOWN]"][c 1][3" "]E[
 c 5]ND[3" "][c 1]D[c 5]ISK[3" "][c 1]T[c
  5]APE[3" "][c 1]P[c 5]RINTER[3" "][c 1]
 R[c 5]UN[c 7]":PRINTTAB(3)                    KO
•810 PRINT"[RVSOFF][c B][LEFT]";:GETA$:IF
 A$=""THENPRINT"[RVSON][c B][LEFT]";:GOTO
 810                                           MF
•820 PRINT" [LEFT]";:IFA$="D"ORA$="T"THEN
 4100                                          AC
•830 IFA$="P"THEN4200                          JE
•840 IFA$="R"THEN900                           HJ
•850 IFA$="E"THENEND                           FI
•860 GOTO810                                   CN
•900 FORI=1TO74:P(I)=0:NEXT:FORI=0TO13:P$
 (I)="":NEXT:H$="":P$="":GOTO200               FA
•950 IF(LEN(X$)+2)>39-POS(0)THENPRINT          OD
•960 PRINTCHR$(32)X$CHR$(Y);:RETURN            AH
•1000 PRINTTAB(21)"[BLACK][BACKARROW]"TAB
 (29)"[RVSON]'[c 5]"                           HA
•1010 PRINT"SCRIPT MAY BE [RVSON][BLACK]V
 ERTICAL[RVSOFF][c 5] OR [c 5][RVSON][BLA
```

```
CK]%&!$%&#[[c 5][RVSOFF] OR"                 MG
•1015 PRINT"IT MAY SLANT TO THE LEFT.":PR
INT                                          AH
•1020 PRINT"SLANT CAN ALSO VARY. FOR INST
ANCE,"                                       OH
•1030 PRINT"SOME LETTERS MAY BE VERTICAL
WHILE"                                       AK
•1050 PRINT"OTHERS ARE SLANTED. IF THE SA
MPLE"                                        BO
•1060 PRINT"HAS MORE THAN ONE SLANT OR TH
E SLANT"                                     CA
•1070 PRINT"APPEARS INCONSISTENT THEN IT'
S"                                           KG
•1080 PRINT"CONSIDERED IRREGULAR.":PRINT     JB
•1085 PRINT"OF THE FOLLOWING CHOSE THE ON
E"                                           MG
•1090 PRINT"THAT CLOSELY MATCHES THE SLAN
T:"K$                                        JB
•1095 GETA$:IFA$=""THEN1095                  KB
•1100 PRINT"[CLEAR][BLUE]"                   JK
•1110 PRINTTAB(17)"3 4 5":PRINTTAB(13)"2[
3" "][c M] [s B] [c G][3" "]6":PRINTTAB(
14)"[s I][3" "][c G][s -][c M][3" "][s U
]"                                           MO
•1120 PRINTTAB(11)"1[3" "][s J]  [s T][s
-][s Y]  [s K][3" "]7":PRINTTAB(12)"[s M
]  [s I]  [s G][s -][s H]  [s U]  [s N]" LP
•1130 PRINTTAB(13)"[s M]  [s J] [3"[s -]"
] [s K]  [s N]":PRINTTAB(14)"[s M] [s I]
 [s H][s -][s G] [s U] [s N]"               KF
•1140 PRINTTAB(15)"[s M] [s J][s Y][s -][
s T][s K] [s N]":PRINTTAB(16)"[s M][s I]
[c M][s -][c G][s U][s N]":PRINTTAB(17)"
[s M][s J][s -][s K][s N]"                   DH
•1150 PRINTTAB(18)"[s M][s -][s N]":PRINT
TAB(8)"[23"[c Y]"]"                          HP
•1160 GOSUB400:IFH$="L"THENA$=MID$("76543
21",VAL(A$),1)                               MI
•1170 GOSUB300:P$(A)=T$(A,V):RETURN          IL
•1200 PRINT"THE BASELINE IS THE IMAGINARY
 LINE THAT"                                  KH
•1210 PRINT"WE WRITE ON. SOME WRITERS HAV
E LINES"                                     JK
•1220 PRINT"THAT ARE STRAIGHT AS A RULER
WHILE"                                       FJ
•1230 PRINT"OTHERS HAVE TROUBLE WRITING A
 STRAIGHT"                                   DN
•1240 PRINT"LINE EVEN WHEN THEY HAVE GUID
E LINES.[DOWN][DOWN][BLUE]"                  PO
•1260 S$="  [c A][5"[s *]"][c S]":FORI=1T
O4:PRINTS$;:NEXT:PRINT                       GF
•1270 S$="[s -][c 4][5"[s C]"][BLUE][s -]
  [s -][c 4][s R][s F][s C][s D][s E][BL
UE][s -]  [s -][c 4][s E][s D][s C][s F]
[s R][BLUE][s -]  [s -][c 4][s C][s F][s
 C][s F][s C][BLUE][s -]"                   EH
•1280 FORI=0TO3:PRINTTAB(2)S$:NEXT           GM
•1290 PRINT"[c 4] ":FORI=0TO3:PRINTTAB(4+
9*I)I+1;:NEXT:PRINT:PRINT                    KB
•1299 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                          DD
•1400 PRINT"THE SIZE OF HANDWRITING IS JU
DGED BY"                                     JN
•1410 PRINT"SIZE OF THE MIDDLE ZONE LETTE
RS SUCH AS"                                  FC
•1420 PRINT"[RVSON][BLACK]A[c 5][RVSOFF],
 [RVSON][BLACK]E[c 5][RVSOFF], [RVSON][B
LACK]M[c 5][RVSOFF], AND [RVSON][BLACK]V
[c 5][RVSOFF]; LETTERS WITHOUT"              EI
•1430 PRINT"UPPER OR LOWER LOOPS. AVERAGE
D-SIZED"                                     AK
•1440 PRINT"HANDWRITNG IS ABOUT 3 ML. HIG
H. SELECT  THE APPROPIATE SIZE:"             IB
•1460 PRINTTAB(9)"[DOWN][DOWN][BLACK][RVS
ON][c D][RVSOFF]  [BACKARROW]":C$=CHR$(2
34)                                          IJ
•1470 PRINTTAB(5)"[RVSON][c +][c M][cEP][
sEP]"C$"  LARGE  [c I][c T][c @][c G]  [
c *][c K]"                                   IP
•1480 PRINTTAB(5)"[RVSON][c Q][RVSOFF]"TA
B(15)"]"                                     CA
•1490 PRINTTAB(7)"[DOWN]1[6" "]2[6" "]3[4
" "]4"                                       LD
•1499 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                          DD
•1600 PRINT"CONTINUITY REFERS TO THE  DEG
REE OF"                                      GO
•1610 PRINT"CONNECTEDNESS BETWEEN LETTERS
. IF THE"                                    LH
•1620 PRINT"WRITING IS MOSTLY OR ENTIRELY
 CONNECTED"                                  HH
•1630 PRINT"THEN PRESS 1. IF IT IS MOSTLY
"                                            IM
•1635 PRINT"OR ENTIRELY DISCONNECTED THEN
 PRESS 2."                                   CC
•1640 PRINT"IF IT IS ABOUT EQUALLY CONNEC
TED AND"                                     OL
•1650 PRINT"DISCONNECTED THEN PRESS 3."      DF
•1660 PRINTTAB(30)"[DOWN][DOWN][BLACK][UP
ARROW]"                                      DF
•1670 PRINT"  [RVSON]CONNECTED  [s Q][s R
]S[s S][s T][s U][s U][s V][s S][s W][s
V][s Q]  [s S][s T]MBI[s W][s R]ON"          BM
•1680 PRINTTAB(7)"[DOWN]1"TAB(19)"2"TAB(3
1)"3[DOWN][DOWN]"                            DH
•1699 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                          DD
•1800 PRINT"THERE ARE A FEW STYLES OF"       CE
•1805 PRINT"WRITING. ROUND SCRIPT APPEARS
 ROUND IN"                                   FC
•1810 PRINT"IN EVERY WAY. THE TOPS OF LET
TERS AND"                                    PM
•1815 PRINT"THE CONNECTING STROKES ARE RO
UND.":PRINT                                  BC
•1820 PRINT"WHEREAS ANGULAR SCRIPT CONTAI
NS LETTERS"                                  GD
```

```
•1825 PRINT"THAT ARE ALL POINTED. OFTEN A
  SAMPLE"                                HN
•1830 PRINT"IS NEITHER ENTIRELY ROUND OR
 ANGULAR"                                GB
•1835 PRINT"BUT A COMBINATION OF BOTH.":P
 RINT                                    LG
•1840 PRINT"INFREQUENTLY, YOU'LL FIND HAN
 DWRITING"                               KL
•1845 PRINT"THAT IS ORGINAL OR ARTISIC: T
 HE"                                     IF
•1847 PRINT"LETTERS ARE UNUSUAL OR UNCONV
 ENTIONAL."                              GN
•1850 PRINT"BUT DON'T MISTAKE SUPERFLUOUS
  LOOPS"                                 JK
•1855 PRINT"AND FLOURISHES AS BEING ORIGI
 NAL."K$                                 EF
•1860 GETA$:IFA$=""THEN1860              LJ
•1870 PRINT"[CLEAR][c 4][3"[DOWN]"]"TAB(1
 8)"[RVSON][s I][RVSOFF]"TAB(26)"+[RVSON]
 '"                                      FP
•1875 PRINTTAB(4)"[RVSON]XOUND"SPC(5)"[s
 B][s C][s D][s E][s F][s B][s G]"SPC(5)"
 *$#&[#["                                FK
•1880 PRINTTAB(16)"[RVSON][s H]":PRINTTAB
 (6)"1"TAB(17)"2"TAB(29)"3":PRINT"[c 4]" BM
•1900 FORB=0TO3:PRINTTAB(7)RIGHT$(STR$(B+
 1),1)". "T$(A,B):NEXT                   KF
•1910 GOSUB405:GOSUB300:P$=T$(A,V):PRINT HB
•1920 PRINT"[DOWN][DOWN][c 5]WHICH BEST D
 ESCRIBES THE HANDWRITING?[c 4][DOWN]"   FI
•1930 FORB=4TO5:PRINTTAB(7)RIGHT$(STR$(B-
 3),1)". "T$(A,B):NEXT                   CK
•1940 GOSUB405:IFV>1THENPRINT"[LEFT]";:GO
 TO1940                                  DC
•1950 A$=CHR$(V+52):GOSUB300:P$(A)=P$+",
 "+T$(A,V+4):RETURN                      FN
•2000 PRINT"LINES CAN BE FAR APART OR SO
 CLOSE"                                  NH
•2010 PRINT"TOGETHER THAT THE LOWER LOOPS
  OF ONE"                                GM
•2020 PRINT"LINE TANGLES WITH THE UPPER L
 OOPS OF"                                PO
•2030 PRINT"THE LINE BELOW:"             JO
•2040 PRINTTAB(13)"[RVSON][BLACK]DRAGGING
 ":PRINTTAB(16)"[s A][PI]  [s A]"        DD
•2050 PRINTTAB(12)"[RVSON]TANGLED[RVSOFF]
  [RVSON]LINES":PRINTTAB(15)"[PI]"       CF
•2060 PRINT"[DOWN][c 5]IF THE LINES ARE V
 ERY TANGLED THEN"                       OM
•2070 PRINT"PRESS 5. IF THE LINES ARE NAR
 ROW BUT"                                NH
•2080 PRINT"ONLY A FEW LOOPS ARE TANGLED
 PRESS 4."                               AJ
•2090 PRINT"IF LINES ARE CLOSE BUT NOT TO
 UCHING"                                 PB
•2100 PRINT"PRESS 3. IF THE LINES ARE WID
 E PRESS 2."                             GC
•2110 PRINT"IF THE SPACE BETWEEN LINES IS
  SO WIDE"                               JN
•2120 PRINT"THAT A ANOTHER LINE COULD BE
 PLACED"                                 FE
•2130 PRINT"BETWEEN THEM THEN PRESS 1.":P
 RINTK$                                  AC
•2140 GETA$:IFA$=""THEN2140              KB
•2199 PRINT"[CLEAR][4"[DOWN]"]":GOSUB400:
 GOSUB300:P$(A)=T$(A,V):RETURN           BD
•2200 PRINT"LEFT AND RIGHT MARGINS ARE TY
 PICALLY"                                HB
•2210 PRINT "ABOUT ONE INCH WIDE. OFTEN T
 HEY ARE"                                KI
•2220 PRINT"MADE WIDER OR NARROWER, OR AR
 E"                                      MC
•2230 PRINT"DIFFERENT WIDTHS. CHOSE ONE T
 HE"                                     BM
•2240 PRINT"FOLLOWING. IF IT DOES NOT FIT
  IN ANY"                                MH
•2250 PRINT"CATEGORY, EG,A MARGIN MAY BE
 NARROW AT"                              NN
•2260 PRINT"AT THE TOP AND WIDE AT THE BO
 TTOM, THEN PRESS 6.":PRINTK$            JP
•2265 GETA$:IFA$=""THEN2265              NL
•2270 PRINT"[CLEAR]"                     HH
•2280 S$="[BLUE][c G][c 4][6"[s C]"][BLUE
 ][c M][3" "][s Y][c 4][7"[s F]"][BLUE][s
  T]  [s B][c 4] [4"[s C]"] [BLUE][s B]":
 FORI=1TO5:PRINTTAB(5)S$:NEXT            NP
•2290 PRINTTAB(8)"[BLACK]1"TAB(20)"2"TAB(
 30)"3[DOWN]"                            KJ
•2300 S$="[BLUE][s B] [c 4][5"[s C]"][BLU
 E][s H][3" "][s H][c 4][6"[s F]"] [BLUE]
 [s B]":FORI=1TO5:PRINTTAB(5)S$:NEXT     FL
•2310 PRINTTAB(8)"[BLACK]3"TAB(20)"4[DOWN
 ]"                                      DP
•2399 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
 URN                                     DD
•2400 PRINT"PRESSURE IS SOMETIMES DIFFICU
 LT TO[6" "]ASSESS. FOR HEAVY";          PM
•2410 PRINT" PRESSURE, TURN THE[4" "]PAGE
  OVER AND SEE OR EVEN FEEL IF THE ";    KD
•2430 PRINT"  PEN HAS LEFT INDENTATIONS I
 N THE PAPER."                           CG
•2440 PRINT"[DOWN]LIGHT PRESSURE APPEARS
 AS DELICATE OR"                         IH
•2450 PRINT"FAINT HANDWRITING. IF PRESSUR
 E APPEARS  TO BE NEITHER HEAVY OR ";    PB
•2470 PRINT"LIGHT[s *][s *]OR YOU[4" "]SI
 MPLY CAN'T TELL[s *][s *]THEN PRESS 3.[D
 OWN][DOWN]"                             IG
•2499 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
 URN                                     DD
•2600 PRINT"SPEED IS ALSO DIFFICULT TO DE
 TERMINE.[3" "]SLOW WRITING IS OFTEN ";  AL
•2610 PRINT"DISCONNECTED AND  CONTAINS AW
 KARD OR FLOURISHED FORMS."              HH
•2650 PRINT"T'S ARE CAREFULLY CROSSED AND
  I'S DOTTEDPRECISELY."                  DO
```

```
•2660 PRINT"[DOWN]FAST SCRIPT IS MOSTLY C
ONNECTED AND[5" "]CONTAINS NATURAL, ";   OE
•2680 PRINT"AND UNCOMPLICATED[5" "]FORMS
 T-BARS AND I-DOTS MAY BE PLACED  TO ";  BK
•2685 PRINT"THE RIGHT. FAST WRITING IS US
UALLY[3" "]INCLINED."                    EL
•2690 PRINT"[DOWN]IF THE WRITING APPEARS
NEITHER FAST OR"                         CC
•2700 PRINT"SLOW OR YOU'RE NOT SURE, THEN
 PRESS 3."                               LL
•2799 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                      DD
•2800 PRINT"UPPER ZONES REFER TO THE UPPE
R LOOPS ON"                              ND
•2805 PRINTTAB(16)"[c 4][UPARROW]  [UPARR
OW][6" "][BACKARROW]"                    MN
•2810 PRINT"[c 5]LETTERS SUCH AS [RVSON][
c 4]B[c 5][RVSOFF], [RVSON][c 4]H[c 5][R
VSOFF], AND [RVSON][c 4]L[c 5][RVSOFF].
THE"                                     KL
•2820 PRINT"LOWER ZONE CONSISTS OF THE LO
WER LOOPS"                               OP
•2830 PRINT"ON LETTERS SUCH AS [RVSON][c
4]G[c 5][RVSOFF], [RVSON][c 4]P[c 5][RVS
OFF], AND [RVSON][c 4]Y[c 5][RVSOFF]."   IA
•2840 PRINTTAB(19)"[c 4]]  [EP][6" "]]"  LP
•2845 PRINT"[c 5]TYPICALLY UPPER AND LOWE
R ZONES ARE[5" "]ABOUT THE SAME HEIGHT O
R";                                      JB
•2846 PRINT" HIGHER THAN THEMIDDLE ZONE.
IF THE ZONES ARE ABOUT[5" "]EQUAL";      OD
•2870 PRINT" PRESS 1. IF THE THE ZONES VA
RY[4" "]THEN PRESS 6.[DOWN]"             KA
•2885 PRINT"IF EITHER UPPER OR LOWER ZONE
 IS SHORT  THEN CONSIDER IT BEING ";     MK
•2895 PRINT"NORMAL. IF BOTH  ZONES ARE SH
ORT THEN PRESS 5."K$                     JL
•2900 GETA$:IFA$=""THEN2900              LF
•2910 PRINT"[CLEAR][c 4]":S$="[RVSON]4044
YGAG"                                    NB
•2920 PRINTTAB(5)"< <<"TAB(15)"> >>"TAB(2
5)"< <<"                                 NM
•2930 FORI=5TO25STEP10:PRINTTAB(I)S$;:NEX
T:PRINT                                  OL
•2940 PRINTTAB(9)"]] ]"TAB(19)"]] ]"TAB(2
9)"[PI][PI] [PI][DOWN]"                  OH
•2950 PRINTTAB(9)"1"TAB(19)"2"TAB(29)"3[D
OWN]"                                    CL
•2960 PRINTTAB(5)">[SS]>>"TAB(25)"<  >"  FF
•2970 FORI=5TO25STEP10:PRINTTAB(I)S$;:NEX
T:PRINT                                  OL
•2980 PRINTTAB(9)"[PI][PI] [PI]"TAB(19)"=
= ="TAB(29)"][PI] =[DOWN]"               IF
•2990 PRINTTAB(9)"4"TAB(19)"5"TAB(29)"6[D
OWN]"                                    CG
•2999 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                      DD
•3000 PRINT"OVALS ARE THE LETTERS [RVSON]
[c 4]A[c 5][RVSOFF] AND [RVSON][c 4]O[RV
SOFF][c 5]. THESE[4" "]ARE USUALLY CLOSE
D:";                                     MO
•3010 PRINT" [RVSON][c 4]OVA4S[RVSOFF][c
5]. BUT SOMETIMESTHEY HAVE OPENINGS AT T
HE TOP: [RVSON][c 4]QV@4S"               LB
•3040 PRINT"[c 5][DOWN]IF MOST OR ALL THE
 OVALS ARE CLOSED"                       FD
•3050 PRINT"THEN PRESS 1. IF MOST OR ALL
OVALS"                                   NB
•3060 PRINT"ARE OPEN PRESS 2. IF THEY ARE
 EQUALLY":PRINT"PRESENT PRESS 3.[DOWN][D
OWN]"                                    IA
•3099 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                      DD
•3200 PRINT"T-BARS ARE THE STROKES THAT C
ROSS THE[3" "]LETTER T. OF THE FOLLOWING
";                                       JM
•3220 PRINT" CHOSE THE[4" "]LENGTH WHICH
IS THE MOST PRAVALENT:"                  BE
•3230 PRINT"[BLACK][DOWN][DOWN]"TAB(10)"[
RVSON]T"TAB(20)"12"TAB(30)"3[DOWN]"      LC
•3235 PRINTTAB(10)"1"TAB(20)"2"TAB(30)"3[
DOWN][DOWN][c 4]"                        LK
•3240 FORB=0TO2:PRINTTAB(7)RIGHT$(STR$(B+
1),1)". "T$(A,B):NEXT                    AG
•3245 GOSUB405:GOSUB300:P$=T$(A,V)+", "  LP
•3250 PRINT"[CLEAR][c 5]T-BARS ALSO VARY
IN POSITION. IF ANY OF THE FOLLOWING APP
EAR";                                    LD
•3255 PRINT" DISTINCTLY AT[6" "]LEAST THR
EE TIMES IN THE SAMPLE THEN"             BH
•3265 PRINT"PRESS 1. IF NOT THEN PRESS AN
Y:[c 4][DOWN][DOWN]"                     NF
•3270 B=3:GOSUB344:PRINT"4":GOSUB333:PRIN
T                                        PE
•3275 B=4:GOSUB344:PRINT"05":GOSUB333:PRI
NT                                       BF
•3280 B=5:GOSUB344:PRINT"62":GOSUB333    LD
•3290 B=6:PRINTTAB(12)"[RVSON]7":GOSUB344
:PRINT"4":GOSUB333:PRINT:B=7:GOSUB344    FO
•3300 PRINT"[RVSON]8":GOSUB333:P$(A)=LEFT
$(P$,LEN(P$)-2):P$="":LL=0:RETURN        IH
•3400 PRINT"I-DOTS ARE INTERPRETED IN A S
IMILAIR WAYTO T-BARS. IF ANY OF THE";    KE
•3420 PRINT" FOLLOWING[6" "]APPEARS DISTI
NCTLY MORE THAN TWICE THEN"              EM
•3430 PRINT"PRESS 1. IF NOT PRESS ANY:[DO
WN][DOWN][c 4]"                          HM
•3450 FORF=0TO5:B=F:GOSUB344:PRINTCHR$(20
2+F):GOSUB333:PRINT                      ND
•3460 NEXT:IFP$=""THENRETURN             OA
•3470 P$(A)=LEFT$(P$,LEN(P$)-2):RETURN   HK
•3600 PRINT"BEGINNING STROKES ARE THE STR
OKES AT[4" "]THE BEGINNING OF WORDS:"    CA
•3620 PRINT"[RVSON][c 4]THE -COMMODORE -C
OMPUTER[RVSOFF]":PRINTTAB(19)"[EP]"      NJ
•3630 PRINT"[c 5]IF MOST OR ALL ALL HAVE
```

```
BEGINNING[7" "]STROKES THEN PRESS 1."     JJ
·3640 PRINT"PRESS 2 IF THEY ARE MOSTLY AB
SENT:"                                    IF
·3650 PRINT"[RVSON][c 4][s W]HE [s S]OMMO
DORE [s S]OMPUTER[RVSOFF]":PRINTTAB(17)"
[EP]"                                     LN
·3660 PRINT"[c 5]IF THEY BOTH APPEAR EQUA
LLY PRESS 3.[3"[DOWN]"]"                  OH
·3699 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                       DD
·3800 PRINT"THE HEIGHT OF CAPITALS IS DET
ERMINED IN"                               OO
·3810 PRINT"RELATIONSHIP TO THE MIDDLE ZO
NE."                                      FP
·3820 PRINT"MEDIUM SIZED CAPITALS ARE ABO
UT TWICE"                                 PG
·3830 PRINT"THE HEIGHT OF MIDDLE ZONE LET
TERS. TALL"                               FA
·3840 PRINT"OR SHORT CAPITALS ARE TALLER
OR SHORTER"                               EK
·3850 PRINT"THEN THE MIDDLE ZONE.[DOWN][D
OWN]"                                     EJ
·3899 GOSUB400:GOSUB300:P$(A)=T$(A,V):RET
URN                                       DD
·4000 PRINT"[DOWN][c 1]D[c 5]ISK OR [c 1]
T[c 5]APE?"                               HH
·4005 GETA$:IFA$=""THEN4005               LF
·4006 IFA$<>"T"ANDA$<>"D"THEN4005         EC
·4010 IFA$="D"THENOPEN2,8,2,F$+",P,R":GOT
04030                                     DM
·4020 OPEN2,1,0,F$                        CO
·4030 INPUT#2,H$,M,N:FORI=0TO14:INPUT#2,P
$:IFI=4ORI=11ORI=12THENGOSUB4050          PH
·4035 P$(I)=P$:NEXT                       PI
·4040 FORI=1TOM:INPUT#2,T1(I):NEXT:FORI=1
TON:INPUT#2,T2(I):NEXT:CLOSE2:GOTO600     NG
·4050 Q$="":FORJ=1TOLEN(P$):M$=MID$(P$,J,
1):IFM$="-"THENM$=","                     GI
·4060 Q$=Q$+M$:NEXT:P$=Q$:RETURN          BA
·4100 PRINT"[c 5][DOWN]":IFA$="D"THENOPEN
2,8,2,"0:"+F$+",P,W":PRINT"[c 4]SAVING "
F$:GOTO4120                               JK
·4110 OPEN2,1,1,F$                        ED
·4120 PRINT#2,H$;N$;M;N$;N:FORI=0TO14:P$=
P$(I):IFI=4ORI=11ORI=12THENGOSUB4150      DN
·4130 PRINT#2,P$:NEXT                     BB
·4140 FORI=1TOM:PRINT#2,T1(I):NEXT:FORI=1
TON:PRINT#2,T2(I):NEXT:CLOSE2:GOTO746     GB
·4150 P$="":FORJ=1TOLEN(P$(I)):M$=MID$(P$
(I),J,1):IFM$=","THENM$="-"               LE
·4160 P$=P$+M$:NEXT:RETURN                FK
·4200 OPEN4,4:PRINT#4,CHR$(14)CHR$(16)"25
SCRIPT ANALYSIS"CHR$(15)                  PN
·4210 PRINT#4,N$N$"NAME: "F$N$N$"HANDWRIT
ING CHARACTERISTICS"                      DL
·4240 FORI=0TO14:PRINT#4,G$(I)": "P$(I):N
EXT:PRINT#4,N$"DOMINANT TRAITS"           IC
·4250 FORI=1TOM-1:PRINT#4,W$(T1(I))CO$;:I
FI/5=INT(I/5)THENPRINT#4                  MB
·4255 NEXT:PRINT#4,W$(T1(M))D$N$          GP
·4270 FORI=1TON-1:PRINT#4,W$(T2(I))CO$;:I
FI/5=INT(I/5)THENPRINT#4                  JF
·4275 NEXT:PRINT#4,W$(T2(N))D$N$          GL
·4280 CLOSE4:GOTO746                      DH
·5000 DATA AESTHETIC,AFFECTIONATE,AGGRESS
IVE,ALOOF,AMBITIOUS                       BO
·5010 DATA AMIABLE,ANXIOUS,ARDENT,ARROGAN
T,ASSERTIVE                               KN
·5020 DATA BALANCED,CALM,CAREFUL,CARELESS
,CAUTIOUS                                 OI
·5030 DATA CONFUSED,CONSCIENTIOUS,CONSERV
ATIVE,CONVENTIONAL,CRITICAL               NI
·5040 DATA DIFFIDENT,DILIGENT,EMOTIONAL,E
NERGETIC,ENTERPRISING                     AF
·5050 DATA EXCITABLE,EXTRAVAGANT,EXTROVER
TED,FASTIDIOUS,FLEXIBLE                   PD
·5060 DATA FRUGAL,GENEROUS,IDEALISTIC,ILL
OGICAL                                    BG
·5070 DATA IMAGINATIVE,IMPATIENT,IMPRESSI
ONABLE,IMPULSIVE,INDECISIVE               CM
·5080 DATA INDEPENDENT,INDIFFERENT,INDIVI
DUALISTIC,INHIBITED,INSINCERE             CH
·5090 DATA INTELLIGENT,INTROVERTED,IRRITA
BLE,LANGUID,LOGICAL                       BN
·5100 DATA LOQUACIOUS,LOYAL,MATURE,METHOD
ICAL,NERVOUS                              EH
·5110 DATA ORGANISED,PASSIONATE,PASSIVE,P
ATIENT,PRACTICAL                          KJ
·5120 DATA REALISTIC,REASONABLE,REFLECTIV
E,RESERVED,RESOLUTE                       IL
·5130 DATA SECLUSIVE,SELF-DISCIPLINED,SEL
FISH,SENSITIVE,SINCERE                    EB
·5140 DATA SPONTANEOUS,TACTFUL,TACTLESS,U
NDISCIPLINED,UNRELIABLE                   BO
·5200 DATA SLANT,7,BASELINE,3,SIZE,3,CONT
INUITY,2,STYLE,5,LINE SPACING,4           FG
·5210 DATA MARGINS,5,PRESSURE,2,SPEED,2,Z
ONES,5,OVALS,2,T-BARS,7,I-DOTS,5          LD
·5220 DATA BEGINNING STROKES,2,CAPITALS,2 GD
·6028 DATA28,160,160,160,160,64,0,0,0     PG
·6029 DATA29,10,10,10,10,10,4,0,0         PM
·6030 DATA30,0,0,0,0,0,0,64,64            LC
·6031 DATA31,0,0,0,0,0,0,16,40            GA
·6033 DATA33,0,0,14,18,32,99,156,0        PM
·6043 DATA43,0,0,0,0,0,2,6,12             EE
·6060 DATA60,0,0,0,0,0,8,8,8              DA
·6061 DATA61,6,10,4,0,0,0,0,0             GA
·6062 DATA62,0,0,8,8,8,8,8,8              BG
·6065 DATA65,10,10,18,50,44,44,56,40      LI
·6067 DATA67,0,0,0,85,170,0,0,0           AH
·6068 DATA68,0,0,85,170,0,0,0,0           DA
·6069 DATA69,0,85,170,0,0,0,0,0           CJ
·6070 DATA70,0,0,0,0,170,85,0,0           GB
·6073 DATA73,8,8,12,4,6,2,3,1             GD
·6074 DATA74,128,128,192,64,96,32,48,16   EP
·6075 DATA75,1,1,3,2,6,4,12,8             HA
```

```
·6077 DATA77,128,192,96,48,24,12,6,3 KG
·6078 DATA78,1,3,6,12,24,48,96,192 GC
·6082 DATA82,0,0,0,0,0,85,170,0 KK
·6085 DATA85,16,16,48,32,96,64,192,128 AP
·6094 DATA94,6,10,10,18,18,18,18,12 HB
·6128 DATA128,0,0,16,36,68,68,187,0 HM
·6129 DATA129,0,0,26,38,66,66,189,0 KE
·6130 DATA130,64,64,92,98,66,67,188,0 AA
·6131 DATA131,0,0,60,98,64,193,62,0 FM
·6132 DATA132,2,2,58,70,66,194,61,0 GK
·6133 DATA133,0,0,28,36,88,96,191,0 MN
·6135 DATA135,0,0,26,38,66,66,189,6 KB
·6136 DATA136,64,64,92,98,66,66,129,0 CI
·6137 DATA137,0,8,0,8,24,40,199,0 OA
·6139 DATA139,64,64,88,100,72,88,135,0 GP
·6140 DATA140,40,40,40,40,40,48,223,0 GC
·6141 DATA141,0,0,84,106,74,74,137,0 AE
·6142 DATA142,0,0,92,98,66,66,129,0 JM
·6143 DATA143,0,0,28,43,82,98,60,0 AH
·6144 DATA144,0,32,44,50,98,162,173,160 KI
·6145 DATA145,0,0,16,35,66,68,184,0 DL
·6146 DATA146,0,0,32,60,36,68,131,0 PF
·6147 DATA147,0,0,12,24,36,68,187,0 FJ
·6148 DATA148,8,127,8,8,24,40,199,0 IH
·6149 DATA149,0,0,66,66,66,70,185,0 EM
·6150 DATA150,0,0,67,66,66,164,24,0 GL
·6152 DATA152,0,0,24,36,68,66,129,0 MK
·6153 DATA153,0,0,66,66,66,70,187,6 JB
·6154 DATA154,0,0,12,18,34,98,156,0 DN
·6155 DATA155,1,1,26,38,68,202,49,0 GA
·6160 DATA160,0,0,0,0,0,0,0,0 AB
·6161 DATA161,0,0,12,18,32,96,159,0 DE
·6162 DATA162,1,1,26,38,68,202,49,0 GE
·6163 DATA163,0,0,28,36,88,97,158,0 CC
·6164 DATA164,10,12,24,48,80,144,15,0 IA
·6165 DATA165,1,0,0,8,24,104,134,0 PJ
·6166 DATA166,0,0,0,44,50,68,135,0 GG
·6167 DATA167,0,0,0,0,3,5,5,10 BN
·6168 DATA168,1,1,26,38,68,202,49,0 CC
·6170 DATA170,12,24,16,38,66,196,56,0 LC
·6173 DATA173,0,0,0,0,0,0,7,0 EK
·6176 DATA176,0,3,0,0,0,0,0,0 NN
·6177 DATA177,8,255,8,8,24,40,199,0 BN
·6178 DATA178,0,192,0,0,0,0,0,0 MH
·6179 DATA179,8,30,8,8,24,40,199,0 FK
·6180 DATA180,8,8,8,8,24,40,199,0 BM
·6181 DATA181,8,232,8,8,24,40,199,0 GL
·6182 DATA182,8,11,8,8,24,40,199,0 NH
·6183 DATA183,0,0,0,0,0,0,63,0 BK
·6184 DATA184,8,8,8,63,24,40,199,0 BG
·6194 DATA194,0,0,3,13,18,38,217,0 FK
·6195 DATA195,0,0,9,27,45,81,160,0 JN
·6196 DATA196,0,0,3,13,18,38,217,6 FC
·6197 DATA197,0,0,17,50,86,154,17,0 MB
·6198 DATA198,4,12,12,20,36,66,129,0 GB
·6199 DATA199,0,0,16,56,73,138,12,0 GH
·6200 DATA200,12,8,16,48,32,0,0,0 NN
·6201 DATA201,0,0,0,0,1,3,2,6 GJ
·6202 DATA202,0,0,16,0,16,16,48,206 GK
·6203 DATA203,0,0,0,0,16,16,48,206 DC
·6204 DATA204,8,0,0,0,16,16,48,206 NJ
·6205 DATA205,0,0,1,0,16,16,48,206 CN
·6206 DATA206,0,0,128,0,16,16,48,206 IH
·6207 DATA207,4,8,0,0,16,16,48,206 FC
·6209 DATA209,2,2,58,70,66,66,60,0 HE
·6210 DATA210,16,0,16,48,80,16,12,0 DO
·6211 DATA211,0,0,28,34,64,66,60,0 PI
·6212 DATA212,0,0,28,34,66,68,56,0 PI
·6213 DATA213,0,0,44,114,34,34,65,0 JM
·6214 DATA214,0,0,28,36,120,66,60,0 OM
·6215 DATA215,16,126,16,16,16,18,12,0 JJ
·6223 DATA223,0,0,0,0,0,93,255,0 DN
·6224 DATA224,0,0,0,0,0,93,255,0 FC
·6225 DATA225,0,0,18,18,18,164,219,0 FJ
·6226 DATA226,0,0,0,0,124,212,87,0 MI
·6227 DATA227,0,0,1,1,205,147,238,0 MK
·6228 DATA228,0,0,32,0,36,109,182,0 PK
·6229 DATA229,0,0,0,0,190,170,201,0 BN
·6230 DATA230,28,34,66,66,66,67,62,10 LM
·6231 DATA231,16,46,34,66,66,130,1,0 EP
·6232 DATA232,28,34,68,88,96,96,159,0 MB
·6233 DATA233,28,38,74,82,98,198,57,0 MJ
·6234 DATA234,24,24,24,24,24,36,195,0 ML
·6235 DATA235,18,18,34,34,34,34,28,0 OI
·6236 DATA236,0,0,0,0,24,24,255,24 BG
·7000 DATAVERY RECLINED,RECLINED,LIGHTLY
RECLINED,VERTICAL,LIGHTLY INCLINED PM
·7002 DATAINCLINED,VERY INCLINED,IRREGULA
R DL
·7010 DATASTRAIGHT,ASCENDING,DESCENDING,U
NEVEN EI
·7020 DATAVERY LARGE,LARGE,MEDIUM,SMALL OE
·7030 DATAMOSTLY CONNECTED,MOSTLY UNCONNE
CTED DD
·7032 DATABOTH CONNECTED & UNCONNECTED BE
·7040 DATAROUND,ANGULAR,ROUND & ANGULAR,O
RIGINAL,LEGIBLE,ILLEGIBLE/MESSY KA
·7050 DATAVERY WIDE,WIDE,MEDIUM,NARROW,TA
NGLED FH
·7060 DATABOTH NORMAL,BOTH NARROW,BOTH WI
DE,WIDE LT/NORMAL OR NARROW RT CF
·7062 DATANARROW OR NORMAL LT/WIDE RT,OTH
ER EG
·7080 DATAHEAVY,LIGHT,MEDIUM,FAST,SLOW,ME
DIUM NC
·7090 DATASYMMETRICAL,EXTENDED UZ,EXTENDE
D LZ IH
·7092 DATABOTH UZ & LZ EXTENDED,BOTH UZ &
 LZ SHORT,IRREGULAR OO
·7100 DATAMOSTLY OPEN,MOSTLY CLOSED,BOTH
OPEN & CLOSED OB
·7110 DATAREGULAR,LONG,SHORT,ABSENT,LEFT,
RIGHT,HIGH,LOW GN
·7120 DATAPRECISELY PLACED,ABSENT,HIGH,RI
GHT,LEFT,DASHED NF
·7130 DATAMOSTLY ABSENT,MOSTLY PRESENT,BO
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering *Ahoy!* programs. Refer to these pages **before** entering any programs!

```
TH ABSENT & PRESENT                                LC
•7140 DATATALL,MEDIUM,SHORT                        EP
•8300 DATA5,39,40,41,44,65,67                      OE
•8301 DATA5,4,39,40,41,44,67,4,39,43,44,4
6,71                                               PI
•8302 DATA7,4,15,31,40,41,49,62,63,5,2,6,
8,19,68,69                                         FD
•8304 DATA7,2,8,23,24,37,56,69,70,6,2,8,2
3,26,38,56,68,3,7,14,16,74                         PI
•8310 DATA6,11,17,29,53,64,66,69,5,5,24,2
5,26,35,70                                         CH
•8312 DATA3,26,39,54,73,5,7,26,39,54,73,7
4                                                  EH
•8320 DATA4,9,22,24,27,28,5,9,27,32,35,40
,42                                                OO
•8322 DATA3,18,55,61,68,9,1,13,18,21,22,2
9,41,45,58,71                                      DO
•8330 DATA6,13,19,22,34,53,61,64,2,34,35,
68,4,1,20,25,42,52                                 OP
•8340 DATA4,1,2,6,30,32,6,3,20,24,36,37,4
5,64                                               EK
•8342 DATA2,1,42,45,3,1,35,42,45,4,13,17,
51,61,66                                           CO
•8344 DATA8,14,16,35,36,37,39,45,54,73             KG
•8350 DATA2,4,63,65,4,13,45,55,71,66,3,11
,28,49,60                                          ME
•8352 DATA6,14,16,31,34,47,70,72,5,14,31,
34,35,63,73                                        MK
•8360 DATA3,11,13,17,55,4,29,50,60,68,72           FM
•8362 DATA2,4,62,65,1,15,21,4,27,29,32,44
,50,0,0                                            CJ
•8370 DATA4,10,24,25,56,64,5,1,21,30,37,4
8,68,0,0                                           DL
•8380 DATA8,1,3,5,10,22,25,30,38,70,8,11,
12,13,18,29,48,57,58,62,0,0                        KE
•8390 DATA3,6,11,12,13,5,5,33,35,37,45,68
,4,5,17,56,59,60                                   HH
•8392 DATA5,5,33,35,37,45,56,3,18,23,59,6
0                                                  KA
•8394 DATA4,14,16,23,37,39                         LO
•8400 DATA5,13,15,41,46,63,71,3,17,43,51,
59,1,51,69                                         CE
•8410 DATA3,11,12,13,66,3,24,36,47,64,2,2
1,57,63                                            HP
•8412 DATA2,14,73,74,1,15,39,2,3,38,72,2,
26,33,35,1,57,58                                   DC
•8420 DATA4,10,17,40,42,52,1,19,53,1,33,3
5                                                  BN
•8430 DATA3,11,13,17,55,2,14,73,74,0,35,2
,17,52,59,2,15,21,39,4,20,35,36,42,47              BK
•8440 DATA3,5,9,10,27,0,40,2,11,30,63              MN
```

BRAINFRAME

FROM PAGE 45

```
•10 DIM R(5),RC(5),N(5),T(15),W$(15),W2$(
15)                                                CC
•20 SC=1024:QA=198:QB=214:WW=54272:GU=0:G
F=0:TT=0:OL=0                                      DB
•30 A$(1)="[RVSON][WHITE]YOU WIN [3"!"][R
VSOFF]":A$(2)="[12" "]"                            HA
•40 A$(3)="[35" "]"                                OJ
•50 WF=54276:P1=54273:VL=54296:AD=54277:F
Q=1:CL=53280:CH=646                                AO
•60 POKECL,3:POKECH,7:POKE53281,0:PRINT"[
CLEAR]"                                            OK
•70 REM                                            JD
•80 FORT=1TO500:F=1:F=0:NEXT                       JI
•90 FORI=1TO6:READH(I):READL(I):READD(I):
NEXT: REM * PITCH VALUES                           LG
•100 FORI=1TO18:PRINT"[HOME][3"[DOWN]"]";
SPC(I);" [s Q]":FORT=1TO80:NEXT:NEXTI              FE
•110 L(1)=225:L(2)=160:R(1)=97:R(2)=160:Q
=1163:POKEQ,160                                    JF
•120 LF=Q-1:RT=Q+1:FORI=1TO10:FORJ=1TO2:P
OKELF,L(J):POKERT,R(J)                             FJ
•130 FORT=1TO60:NEXT:NEXTJ:LF=LF-1:RT=RT+
1:NEXTI                                            EP
•140 FORT=1TO200:NEXT:MM$="BRAINFRAME":Q=
LF+2                                               OK
•150 FORI=1TO10:M$=MID$(MM$,I,1):M=ASC(M$
):POKEQ,M+128-64                                   JO
•160 FQ=FQ+10:GOSUB1220:Q=Q+2:FORT=1TO300
:NEXT:NEXTI                                        AD
•170 FORT=1TO400:FQ=FQ+10:GOSUB1220:PRINT
"[c 6]"                                            HI
•180 PRINT"[DOWN][DOWN]";TAB(9);"AN EXCER
CISE IN LOGIC"                                     EI
•190 PRINT"[DOWN][DOWN]";TAB(13);"FOR THE
 C-64[DOWN][DOWN][7"[LEFT]"]BY"                    AG
•200 PRINT"[DOWN][DOWN]";TAB(14);"NORM VO
GEL"                                               ML
•210 FORT=1TO425:F=1:F=0:F=T/33:NEXT               DJ
•220 PRINT"[HOME][6"[DOWN]"]";:GOSUB1260           BA
•230 PRINT"[PURPLE] I WILL SELECT A 3 TO
5 - DIGIT NUMBER[3" "](YOUR CHOICE),";             DI
•240 PRINT" COMPOSED OF NUMBERS FROM [WHI
TE]0 - 9.[PURPLE]  (NOTE: EACH NUMBER";            GC
•250 PRINT" CAN BE USED  [WHITE]MORE THAN
 ONCE).[PURPLE]  AFTER ENTERING YOUR[3"
"]TRY";                                            KN
•260 PRINT" (YOU HAVE 15), I WILL TELL YO
U HOW  MANY NUMBERS ARE";                          HI
•270 PRINT" CORRECT & IN THE [WHITE]COR-
RECT POSITION,";                                   FH
•280 PRINT" [PURPLE]AND HOW MANY ARE CORR
ECT BUT IN THE [WHITE]WRONG[PURPLE] POSI
TION."                                             JG
•290 PRINT"[DOWN] IF YOU GIVE UP DURING T
HE GAME, JUST[4" "]TYPE IN";                       MM
•300 PRINT" [WHITE]E[PURPLE], AND I'LL TE
LL YOU WHAT THE[3" "]SOLUTION WAS AND";            BN
```

```
•310 PRINT" END THE GAME.":PRINTTAB(13);"
[DOWN][WHITE]GOOD LUCK!"                   BP
•320 PRINT"[DOWN] ENTER THE SIZE OF THE N
UMBER (3/4/5)";                            FJ
•330 GETND$:IFND$=""THEN330                BG
•340 IFASC(ND$)<51ORASC(ND$)>53THEN330     BF
•350 ND=VAL(ND$):GOSUB1260                 JF
•360 IFND<>OLTHENGF=0:TT=0:GU=0:FORI=1TO1
5:W$(I)=" ":W2$(I)=" ":NEXT                LJ
•370 PRINT"[CLEAR][DOWN]";TAB(10);"[RVSON
]";ND;" NUMBERS[RVSOFF]"                   IN
•380 PRINT"[DOWN][DOWN]";TAB(28);"NO.[3"
"]WRONG"                                   KP
•390 PRINT" TRY";TAB(27);"[CYAN]&[WHITE]
POS[3" "]POS"                              EI
•400 POKEQB,22:PRINT:PRINT" ([YELLOW]'R'
[WHITE]TO RE-DO LINE / [YELLOW]'E'[WHITE
] TO END GAME)"                            DM
•410 TT=TT+1:TW=0:SP=10:P=9-ND:SP=10:IFND
=3THENSP=12                                NI
•420 FORI=1TOND:R(I)=INT(RND(0)*10):NEXT:
BC=7:BB=1                                  EM
•430 POKEQB,6:PRINT:                       JE
•440 REM                                   JD
•450 REM  *** START OF PLAY LOOP: ***      IJ
•455 REM                                   JD
•460 TW=0:YG=0:FORZ=1TO15:PRINTA$(3)       BE
•470 PRINT "[UP][c 3]";Z;:POKECH,BC+BB:W=
6                                          MO
•480 FORI=1TOND                            PM
•490 GETN$:IFN$=""THEN490                  LK
•500 IFN$="E"THENGOSUB1370:GOTO870         MN
•510 IFN$="R"THENGOSUB1260:PRINT:PRINT"[U
P]";SPC(3);A$(3):GOTO470                   DJ
•520 IFASC(N$)<48ORASC(N$)>57THEN490       BP
•530 PRINTTAB(W);N$;:N(I)=VAL(N$):W=W+P:N
EXTI                                       KC
•540 FORI=1TOND:RC(I)=R(I):FQ=N(I)+25:GOS
UB1220                                     EH
•550 POKE53280,N(I)+1:FORJ=1TO450:NEXTJ:N
EXTI                                       LJ
•560 REM                                   JD
•570 REM  ***  CORPOS COMPARATOR:  ***     NM
•580 CP=0:FORI=1TOND:IFN(I)=RC(I)THENCP=C
P+1:N(I)=98:RC(I)=99                       ME
•590 NEXTI:IFCP=ND-1THENTW=TW+1            LC
•600 REM                                   JD
•610 REM  ***  COR COMPARATOR:  ***        OL
•620 CO=0:FORI=1TOND:FORJ=1TOND:IFN(I)=RC
(J)THENCO=CO+1:N(I)=98:RC(J)=99            LF
•630 NEXTJ:NEXTI                           NB
•640 IFCP=NDTHEN740                        KA
•650 REM  ***  IN-GAME PROMPTS:  ***       OM
•660 IFCP<>NDTHEN PRINTTAB(28);CP;"[4" "]
";CO:BB=-BB                                CD
•670 IFND=3ANDZ=10THENR=4:GOSUB1070        KH
•680 IFTW=4THENR=2:GOSUB1070               LP
•690 IFCP+CO=NDANDYG=0THENR=1:GOSUB1070    OF
```

```
•700 IFZ=14THENR=3:GOSUB1070               IG
•710 IFZ=15THENR=5:GOSUB1070:GOTO940       OB
•720 NEXTZ                                 NM
•730 REM  ***  WIN ROUTINE:  ***           OL
•740 F=2:G=1:POKECL,2:PRINT"[CLEAR][DOWN]
[DOWN]";TAB(5);A$(1);SPC(7);Z;" TRIES"     EJ
•750 GOSUB1310:PRINT"[DOWN][DOWN]";TAB(12
);"[PURPLE]THE ANSWER WAS:[DOWN][DOWN][W
HITE]"                                     DC
•760 PRINTSPC(SP);:FORI=1TOND:PRINTR(I);S
PC(6-ND);:NEXT:PRINT                       FK
•770 T(Z)=T(Z)+1:T$=STR$(T(Z)):W$(Z)=MID$
(T$,2,1):IFT(Z)<10THEN790                  OL
•780 W2$(Z)=RIGHT$(T$,1)                   OJ
•790 IFZ>NDGOTO820                         EL
•800 PRINT"[DOWN][DOWN][CYAN]UNBELIEVEABL
E! 'GIVEUPS' REDUCED TO '0'!":GU=0:GF=1    HM
•810 GOTO940                               DB
•820 IFZ<=ND+2THENPRINT"[DOWN][DOWN][CYAN
]NEXT TIME, I'LL HAVE TO GIVE U A HARDER
 ONE!":GOTO940                             IN
•830 IFZ<ND+6THENPRINT"[DOWN][DOWN][CYAN]
  NOT BAD, BUT U NEED MORE PRACTICE!":GO
TO940                                      ME
•840 PRINT"[DOWN][CYAN]YOU NEED ALOT MORE
 PRACTICE! TRY AGAIN!":GOTO940             CI
•850 REM                                   JD
•860 REM ***  GIVEUP ROUTINE:  ***         LD
•870 GU=GU+1:IFGF=1THENGU=0                FB
•880 POKECL,2:PRINT"[CLEAR][3"[DOWN]"]";T
AB(12);"THE ANSWER WAS:[DOWN][DOWN][WHIT
E]"                                        LO
•890 PRINTSPC(SP);:FORI=1TOND:PRINTR(I);S
PC(6-ND);:NEXT:PRINT                       FK
•900 PRINT"[DOWN][DOWN][CYAN] TRY AGAIN!
YOU MIGHT GET IT NEXT TIME!"               BC
•910 F=2:G=0                               JM
•920 REM                                   JD
•930 REM  ***  STATS:  ***                 FI
•940 PRINT:PRINT"[DOWN][c 7]TRIES:  1 2 3
 4 5 6 7 8 9 0 1 2 3 4 5"                  FA
•950 PRINT"  WON:[c 6]  ";:FORI=1TO15:PRI
NTW$(I);" ";:NEXT                          NK
•960 PRINT:PRINTTAB(8);:FORI=1TO15:PRINTW
2$(I);" ";:NEXT                            IP
•970 PRINT:PRINT"[DOWN][c 3][4" "]TOTAL G
AMES: ";TT;SPC(3);"GIVEUPS:";GU            CO
•980 PRINT"[DOWN][WHITE][5" "]DO YOU WANT
 TO TRY AGAIN?"                            DL
•990 PRINT"  ([YELLOW]Y/N[WHITE] OR [YELL
OW]I[WHITE] TO CHANGE # OF DIGITS)";       GD
•1000 GETYN$:PRINT"[HOME][DOWN][DOWN]";TA
B(5);A$(F):F=F-G                           GF
•1010 IFYN$="Y"THENFQ=150:GOSUB1220:GOTO3
70                                         GI
•1020 IFYN$="I"THENPRINT"[CLEAR][3"[DOWN]
"]":GOSUB1260:OL=ND:GOTO230                LB
•1030 IFYN$<>"N"THENG=-G:FORT=1TO400:NEXT
```

IMPORTANT! Letters on white background are **Bug Repellent** line codes. **Do not enter them!** Pages 85 and 86 explain these codes and provide other essential information on entering ***Ahoy!*** programs. Refer to these pages **before** entering any programs!

```
:GOTO1000                                              KN
·1040 PRINT"[CLEAR]":END                               CI
·1050 REM                                              JD
·1060 REM  *** IN-GAME PROMPTS:                        JJ
·1070 ONRGOTO1080,1100,1120,1130,1150                  DG
·1080 FQ=200:GOSUB1220:PRINTTAB(6);"[YELL
OW]YOU'VE GOT 'EM ALL NOW!"                            BD
·1090 YG=1:GOTO1140                                    CK
·1100 GOSUB1370:PRINTTAB(8);"[YELLOW]GOIN
G FOR BROKE, EH?":TW=99:GOTO1140                       LH
·1120 FQ=0:GOSUB1220:PRINTTAB(6);"[YELLOW
]THIS IS YOUR LAST CHANCE!":GOTO1140                   HB
·1130 GOSUB1370:PRINTTAB(8);"[YELLOW]YOU
HAVEN'T GOT IT [CYAN]YET?!"                            PA
·1140 PRINT"[UP]";:FORT=1TO350:X=T/33:NEX
T:RETURN                                               PA
·1150 GOSUB1370:PRINT:PRINT"[5" "][YELLOW
]GOOD GUESS, BUT THE ANSWER WAS:[4" "][W
HITE]"                                                 MP
·1160 PRINTA$(3)                                       JG
·1170 PRINT"[UP][UP]";SPC(SP);:FORI=1TOND
:PRINTR(I);SPC(6-ND);:NEXT:PRINT                       KE
·1180 POKEQA,0:PRINT"[8" "][c 3]HIT ANY K
EY TO CONTINUE [s Z]":WAITQA,1                         IK
·1190 PRINT"[CLEAR][4"[DOWN]"]":GOSUB1260
:F=2:G=0:POKEQA,0:RETURN                               LO
·1200 REM                                              JD
·1210 REM  ***  PITCHES:                               FM
·1220 POKEVL,14:POKEWF,0:POKEP1,FQ+20:POK
EAD,25:POKEWF,17                                       LN
·1230 FORT=1TO100:NEXT:POKEWF,0:RETURN                 PN
·1240 REM                                              JD
·1250 REM  ***  SWEEP:                                 NH
·1260 FORI=1TO22:POKEWW+I,0::NEXTI                     NF
·1270 POKEWW+24,12:POKEWW+5,80:POKEWW+6,2
43:POKEWW+3,4::POKEWW+4,129                            AI
·1280 FORI=0TO140STEP4:POKEWW+1,I:NEXTI:P
OKEWW+4,128:FORI=1TO50:NEXTI                           JH
·1290 POKEWW+4,128:RETURN                              KH
·1300 REM ***  TUNE:                                   IB
·1310 FORI=WWTOWW+24:POKEI,0:NEXT                      AI
·1320 POKEVL,6:POKEAD,31:POKEWW+3,200:POK
EWW+2,200:POKEWW+6,89                                  BO
·1330 FORI=1TO6                                        JB
·1340 POKEWW+1,H(I):POKEWW,L(I):POKEWF,65              BA
·1350 FORT=1TOD(I):NEXT:POKEWF,64:FORT=1T
010:NEXT:NEXTI:RETURN                                  CO
·1360 REM ***  RAZZ:                                   KG
·1370 POKEVL,15:POKEAD,45:POKEAD+1,165                 KC
·1380 POKEWF,33:POKEWF-3,6:POKEWF-4,5                  BA
·1390 FORT=1TO600:NEXT:POKEWF,32:POKEWF-3
,0:POKEWF-4,0:RETURN                                   HD
·1400 DATA 22,96,120,25,30,120,28,49,120               MG
·1410 DATA 33,135,325,28,49,110,33,135,10
00                                                     HJ
```

PRINTAT

FROM PAGE 45

```
·10 REM**SET UP THE PRINTAT ROUTINE**                  BE
·12 AT=679:FORN=ATTOAT+22:READM:POKEN,M:N
EXT                                                    DJ
·14 DATA32,180,2,132,251,32,180,2                      AN
·16 DATA166,251,76,240,255,32,253,174                  KO
·18 DATA32,158,173,32,170,177,96                       PI
·20 REM**EXAMPLE PROGRAM USING PRINTAT**               GA
·22 POKE53280,14:POKE53281,12                          LM
·24 PRINTCHR$(147)CHR$(144)                            BM
·26 SYSAT,2,0:PRINT"NOW"                               CK
·28 SYSAT,5,2:PRINT"YOU"                               DG
·30 SYSAT,8,4:PRINT"CAN"                               PI
·32 SYSAT,11,6:PRINT"PRINT"                            KL
·34 SYSAT,8,17:PRINT"NEATLY"                           AP
·36 SYSAT,11,16:PRINT"FORMATTED"                       NN
·38 SYSAT,14,17:PRINT"SCREENS"                         EP
·40 FORN=0TO2*[PI]STEP[PI]/40                          AP
·42 L=INT(12+SIN(N)*8)                                 EB
·44 C=INT(20+COS(N)*8)                                 FO
·46 SYSAT,L,C:PRINT"*":NEXT                            AL
·48 SYSAT,22,0                                         DE
```

LUCKY LOTTERY

FROM PAGE 53

```
·30 L$="[HOME][23"[DOWN]"]"                            LK
·40 GOSUB 1000:GOTO 1500                               NO
·50 DIM NUM(45),TN(NN),PF(NN):C=0:B=0                  ML
·70 PRINT "[CLEAR]":POKE 53280,6:POKE 532
81,1                                                   OI
·80 PRINT LEFT$(L$,2)TAB(8)"[c 4][c A][21
"[s *]"][c S]"                                         ML
·90 PRINT LEFT$(L$,3)TAB(8)"[s B][21" "][
s B]"                                                  ED
·100 PRINT LEFT$(L$,4)TAB(8)"[s B]  [RED]
LUCKY LOTTO MENU[c 4][3" "][s B]"                      HM
·120 PRINT LEFT$(L$,5)TAB(8)"[s B][21" "]
[s B]"                                                 MJ
·125 PRINT LEFT$(L$,6)TAB(8)"[c Z][21"[s
*]"][c X]"                                             ED
·130 PRINT LEFT$(L$,10)TAB(8)"[BLUE][RVSO
N][1][RVSOFF] -LOAD NUMBER FILE"                       LE
·140 PRINT LEFT$(L$,11)TAB(8)"[RVSON][2][
RVSOFF] -ENTER CURRENT NUMBERS"                        PM
·150 PRINT LEFT$(L$,12)TAB(8)"[RVSON][3][
RVSOFF] -DISPLAY ALL NUMBERS"                          IN
·160 PRINT LEFT$(L$,13)TAB(8)"[RVSON][4][
RVSOFF] -6 MOST COMMON NUMBERS"                        NM
·170 PRINT LEFT$(L$,14)TAB(8)"[RVSON][5][
RVSOFF] -END SESSION"                                  CE
·175 PRINT LEFT$(L$,20)TAB(10)"[RED]YOUR
CHOICE ? [RVSON][BLUE][1-5][RVSOFF]"                   NO
```

```
•180 GET AN$:IF AN$="" THEN180                CH
•190 AN=VAL(AN$)                              BK
•200 ON AN GOSUB 730,220,310,420,660          GG
•210 REM ========================             JO
•211 REM ****  INPUT NUMBERS  ****            ML
•212 REM ========================             JO
•220 PRINT "[CLEAR]":POKE 53280,12:POKE 5
 3281,1                                       LM
•230 PRINT TAB(4)"[RED][DOWN][s U][30"[s
 *]"][s I]"                                   DB
•240 PRINT TAB(4)"[RED][RVSON] ENTER THIS
  WEEKS LOTTO NUMBERS "                       OP
•250 PRINT TAB(4)"[RED][DOWN][UP][s J][30
 "[s *]"][s K]"                               AA
•260 PRINT LEFT$(L$,8):T=6                    CB
•270 FOR A=1 TO 6:PRINT TAB(10)"NUMBER ";
 A;:INPUT N(A)                                EJ
•280 FOR S=1 TO NN:IF N(A)=S THEN630          NO
•290 NEXT:NEXT                                EF
•301 GOTO70                                   PF
•304 REM ========================             JO
•305 REM **** DISPLAY NUMBERS ****            GG
•306 REM ========================             JO
•310 PRINT "[CLEAR]":POKE 53280,3:POKE 53
 281,1:T=4                                    JJ
•320 PRINT TAB(6)"[UP][RED][RVSON] CURREN
 T LOTTO NUMBER LIST "                        BM
•330 FOR S=1 TO NN:IF S=23 THENPRINT "[HO
 ME]":T=24                                    EJ
•350 IF S>9 THEN370                           BE
•360 PRINT "[BLUE]"TAB(T);S" "TN(S):GOTO3
 80                                           LK
•370 PRINT "[BLUE]"TAB(T-1);S" "TN(S)         DH
•380 NEXT                                     IA
•390 PRINTLEFT$(L$,25)"[c 4]  PRESS [RVSO
 N] F1 [RVSOFF] FOR 6 MOST COMMON NUMBERS
 ";                                           PB
•400 GET AN$:IF AN$="" THEN400                DC
•410 IF AN$<>"[F1]" THEN400                   NC
•414 REM ========================             HL
•415 REM **** COMMON NUMBERS ****             IC
•416 REM ========================             HL
•420 FOR I=1 TO NN:IF TN(I)>B THENB=TN(I)     NF
•430 NEXT                                     IA
•445 PRINT"[CLEAR]":POKE 53280,6:POKE 532
 81,1                                         OI
•450 PRINT LEFT$(L$,1)TAB(8)"[RED][RVSON]
  6 MOST COMMON NUMBERS "                     II
•460 PRINT LEFT$(L$,3)TAB(12)"NUMBER";TAB
 (22)"TIMES"                                  PN
•470 FOR E=1 TO 6:SW=0                        GO
•480 IF B<=0 THEN540                          LK
•490 FOR I=1 TO NN:T=14                       LD
•495 IF I>9 THEN T=13                         ML
•500 IF TN(I)=B THEN PRINT TAB(T);I;SPC(7
 );TN(I):SW=1                                 MM
•510 NEXT                                     IA
•520 IF SW=0 THENB=B-1:GOTO480                FM
```

```
•530 B=B-1                                    BA
•540 NEXT E                                   MJ
•550 PRINTLEFT$(L$,22)TAB(5)"PRESS [RVSON
 ][BLACK]F1[RVSOFF][RED] TO ENTER MORE NU
 MBERS"                                       GF
•560 PRINT TAB(5)"PRESS [RVSON][BLACK]F3[
 RVSOFF][RED] TO VIEW ALL NUMBERS"            FC
•570 PRINT TAB(5)"PRESS [RVSON][BLACK]F5[
 RVSOFF][RED] FOR MAIN MENU";                 GK
•580 GET A$:IF A$="" THEN580                  JI
•590 IF A$="[F1]" THEN220                     KJ
•600 IF A$="[F3]" THEN310                     KK
•610 IF A$="[F5]" THEN 70                     IP
•620 IF A$<>"[F1]" OR A$<>"[F3]" OR <>"[F
 5]" THEN580                                  JL
•630 TN(S)=TN(S)+1:GOTO290                    HF
•640 REM ========================             HL
•650 REM ****  WRITE ROUTINE ****             MA
•660 REM ========================             HL
•670 OPEN 2,8,2,"@0:"+NF$+",S,W"              OA
•680 FOR S=1 TO NN                            PE
•690 PRINT#2,TN(S)                            EH
•700 NEXT                                     IA
•710 CLOSE 2:END                              JB
•715 REM ========================             HL
•720 REM ****  READ ROUTINE  ****             CO
•725 REM ========================             HL
•730 PRINT"[CLEAR]":POKE 53280,6:POKE 532
 81,6                                         OP
•735 PRINT LEFT$(L$,10)TAB(10)"[WHITE]LOA
 DING NUMBER FILE[3"."]"                      HF
•736 PRINT LEFT$(L$,12)TAB(11)"ONE MOMENT
  PLEASE"                                     BC
•740 OPEN 2,8,2,"@0:"+NF$+",S,R"              OJ
•750 FOR S=1 TO NN                            PE
•760 INPUT#2,TN(S)                            CD
•770 NEXT S                                   NH
•780 CLOSE 2:GOTO 70                          AA
•990 REM ==========================           ME
•991 REM ****  SPRITE MOVEMENT ****           NO
•992 REM ==========================           ME
•1000 FOR S=50880 TO 51116:READ A:POKE S,
 A:NEXT                                       OD
•1010 DATA 169,255,45,0,198,240,16,169,0,
 141,0,198,162,21,189,0                       MN
•1020 DATA 197,157,0,198,202,208,247,162,
 1,169,1,141,80,197,173,80                    KK
•1030 DATA 197,45,0,197,240,3,76,243,198,
 232,232,14,80,197,208,238                    OJ
•1040 DATA 76,49,234                          FG
•1050 DATA 169,0,29,0,197,208,3,76,97,199
 ,169,128,61                                  DG
•1060 DATA 0,197,240,48,254,0,198,208,40,
 222,255,207,76,144,199,80                    LE
•1070 DATA 197,45,16,208,208,12,173,16,20
 8,13,80,197,141,16,208,76                    NB
•1080 DATA 43,199,173,16,208,77,80,197,14
 1,16,208,189,0,197,157,0                     OO
```

```
·1090 DATA 198,76,97,199,222,0,198,208,40
,254,255,207,208,29,173,80                    AB
·1100 DATA 197,45,16,208,208,12,173,16,20
8,13,80,197,141,16,208,76                     NB
·1110 DATA 91,199,173,16,208,77,80,197,14
1,16,208,189,0,197,157,0                      CD
·1120 DATA 198,169,0,232,29,0,197,208,3,7
6,140,199,169,128,61,0                        PO
·1130 DATA 197,240,11,254,0,198,208,20,22
2,255,207,76,134,199,222,0                    KC
·1140 DATA 198,208,9,254,255,207,189,0,19
7,157,0,198,202,76,233,198                    FP
·1150 DATA 169,255,221,255,207,240,3,76,4
3,199,173,80,197,76,17,199                    BD
·1160 DATA 120,169,192,141,20,3,169,198,1
41,21,3,88,96                                 JL
·1170 SP=53248:SS=50433:PRINT "[CLEAR]":P
OKE 53280,9:POKE 53281,0                      NP
·1180 H=1:V=1                                 JF
·1190 FOR L=SS TO SS+16 STEP 2                FH
·1200 POKE L,H:POKE L+1,V                     OC
·1210 H=INT(RND(0)*247)+1:V=INT(RND(0)*3)
+1:NEXT                                       CH
·1220 POKE 50688,FF                           FO
·1230 FOR S=2040 TO 2047                      MK
·1240 POKE S,14:NEXT                          EA
·1250 FOR S=896 TO 959:READ CH                NE
·1260 POKE S,CH:NEXT                          HG
·1270 S=1                                     DL
·1280 FOR L=53287 TO 53293                    CE
·1290 POKE L,S:S=S+1:NEXT:POKE 53294,14       LO
·1310 S=20:A=30                               PN
·1320 FOR L=53248 TO 53263 STEP 2             PG
·1330 POKE L,S+48:POKE L+1,A+40               OF
·1340 S=S+25:A=A+25:NEXT                      KB
·1350 POKE 53269,255                          FI
·1360 POKE 50432,255                          HF
·1370 SYS 51104                               KF
·1380 RETURN                                  IM
·1384 REM ======================              PM
·1385 REM ****  SPRITE DATA ****              LF
·1386 REM ======================              PM
·1390 DATA 1,129,128,63,255,252,127,255,2
54,255,255,255,255,255,255                    IO
·1400 DATA 241,129,128,241,129,128,241,12
9,128,255,255,252,255,255,254                 GI
·1410 DATA 127,255,255,63,255,255,1,129,1
43,1,129,143,1,129,143,255,255,255,255        NH
·1420 DATA 255,255,127,255,254,63,255,252
,1,129,128,1,129,128,0                        FG
·1500 PRINT:PRINT:PRINT TAB(4)"[c 6][RVSO
N]$[RVSOFF][11" "][RVSON]$[RVSOFF]"           BG
·1510 PRINT TAB(4)"[RVSON]$[RVSOFF][11" "
][RVSON]$[RVSOFF]"                            BL
·1520 PRINT TAB(4)"[RVSON]$[RVSOFF][3" "]
[RVSON] [RVSOFF] [RVSON] [RVSOFF][5" "][
RVSON]$[RVSOFF][3" "][RVSON][3" "]"           LP
·1530 PRINT TAB(4)"[RVSON][3"$"][RVSOFF]
[RVSON] [RVSOFF] [RVSON] [RVSOFF][5" "][
RVSON][3"$"][RVSOFF] [RVSON] [RVSOFF] [R
VSON] "                                       DD
·1540 PRINT TAB(8)"[RVSON] [RVSOFF] [RVSO
N] [RVSOFF] [RVSON][3" "][RVSOFF][5" "][
RVSON] [RVSOFF] [RVSON] [RVSOFF] [RVSON]
[3" "]"                                       AL
·1550 PRINT TAB(8)"[RVSON][3" "][RVSOFF]
[RVSON] [RVSOFF][7" "][RVSON][3" "][RVSO
FF]  [RVSON] "                                BC
·1560 PRINT TAB(12)"[RVSON] [RVSOFF][3" "
][RVSON] [sEP] [RVSOFF][6" "][RVSON] [RV
SOFF]  [RVSON][3" "]"                         BB
·1570 PRINT TAB(12)"[RVSON][3" "][RVSOFF]
 [RVSON]  [RVSOFF][sEP][6" "][RVSON] [RV
SOFF][3" "][RVSON] "                          PB
·1580 PRINT TAB(16)"[RVSON]  [c *][RVSOFF
] [RVSON] [RVSOFF] [RVSON] [RVSOFF][6" "
][RVSON] [RVSOFF]  [RVSON][3" "]"             GD
·1590 PRINT TAB(16)"[RVSON] [RVSOFF][c *]
[RVSON] [RVSOFF] [RVSON][3" "][RVSOFF][6
" "][RVSON] [RVSOFF]  [RVSON] [RVSOFF] [
RVSON] "                                      MN
·1600 PRINT TAB(21)"[RVSON] [RVSOFF][10"
"][RVSON] [RVSOFF] [RVSON] "                  CL
·1610 PRINT TAB(21)"[RVSON] [RVSOFF][10"
"][RVSON][3" "]"                              IH
·1620 PRINT:PRINT "[WHITE][40"[c P]"]"        PO
·1630 PRINT "[UP][RVSON][c 2][18" "][BLAC
K] [c 2][21" "]"                              DO
·1640 PRINT "[UP][RVSON][11" "][WHITE]DOU
BLE[c 2] [BLACK]  [c 2] [WHITE]SOFTWARE[
c 2][12" "]"                                  LK
·1650 PRINT "[UP][RVSON][19" "][BLACK]  [
c 2][19" "]"                                  IH
·1660 PRINT "[UP][WHITE][40"[c Y]"]"          MC
·1670 PRINT TAB(10)"[UP][c 2]DESIGNED BY[
3"."]"                                        PE
·1680 PRINT TAB(19)"BOB LLORET"               NL
·1690 PRINT:PRINT TAB(11)"[WHITE]PRESS [R
ED]F1 [WHITE]TO START";                       IA
·1700 GET A$:IF A$="" THEN 1700               LG
·1710 IF A$="[F1]" THEN FOR A=SP TO SP+7:
POKE SP+21,0:NEXT:GOTO 2000                   CG
·1720 IF A$<>"[F1]" THEN 1700                 JG
·2000 PRINT"[CLEAR]"                          HH
·2010 PRINTLEFT$(L$,10)" ENTER YOUR FILE
NAME";:INPUT NF$                              DF
·2015 IF NF$="" THEN 2010                     LA
·2020 PRINTLEFT$(L$,12)" ENTER HIGHEST NU
MBER IN YOUR LOTTO";:INPUT NN                 BH
·2025 IF NN<1 THEN 2020                       IE
·2030 GOTO50                                  PD
```

ALL THE PROGRAMS IN THIS ISSUE OF *AHOY!* ARE AVAILABLE ON DISK OR CASSETTE. SEE PAGE 65 FOR DETAILS.

SMALL THINGS CONSIDERED

Our congratulations to the following Commodore users, winners of subscriptions to *Ahoy!* courtesy of New York's Small Things Considered radio show (heard weeknights 5-8 and Saturdays 6-8 on WNYC AM83):

Thomas Louie Jeremy Rubinschneider Susie Freese Mario Costonz Misty Longo

S.O.S. *Continued from page 74*

removing the RS232 card since I will interchange the computers with the same printer) with a C-64. Is there a hardware product to do this, preferably one with few or no software commands?

–Todd Walton
Vicksburg, MS

You need an RS232 interface for your Commodore. This is a relatively inexpensive device that can be easily found at most stores that handle Commodore products. In general there are two different types of interfaces: terminal type and printer type. You need the printer type. You may then use the printer as any other one by simply changing one number when opening the printer channel for output.

...COMING IN THE AUGUST *AHOY!* (ON SALE JULY 2)...

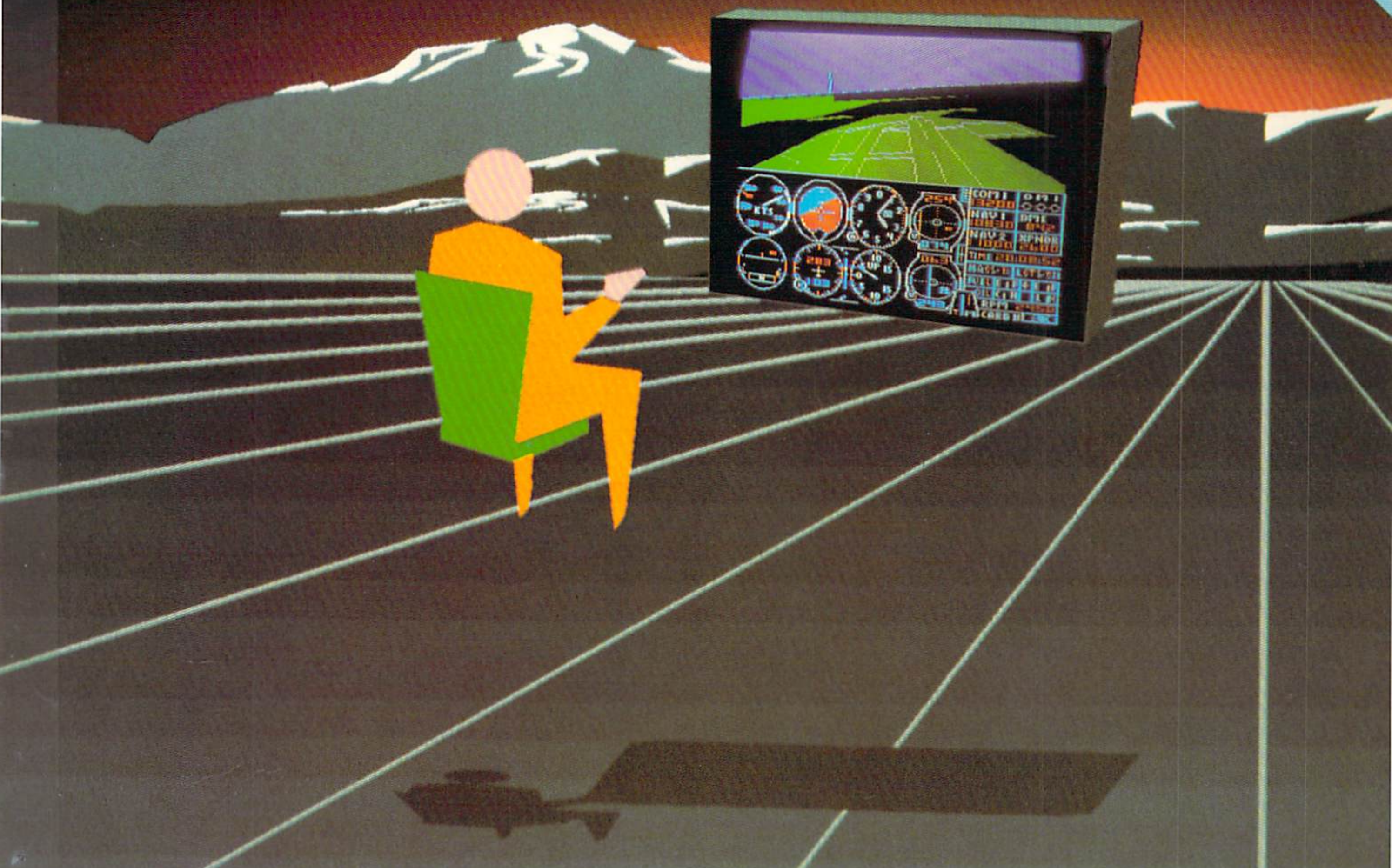
WINDOWS ON THE VIC AND 64
ADD MACINTOSH MAGIC TO YOUR COMMODORE PROGRAMS

OPENING CEREMONIES FOR THE 128
COMMODORE'S NEW COMPUTER FROM THE INSIDE OUT

FALLING NEEDLES AND SHARED BIRTHDAYS
SIMULATIONS OF THE REAL WORLD

Flight Simulator II

For the Commodore 64™



Put yourself in the pilot's seat of a Piper 181 Cherokee Archer for an awe-inspiring flight over realistic scenery from New York to Los Angeles. High speed color-filled 3D graphics will give you a beautiful panoramic view as you practice takeoffs, landings, and aerobatics. Complete documentation will get you airborne quickly even if you've never flown before. When you think you're ready, you can play the World War I Ace aerial battle game. Flight Simulator II features include ■ animated color 3D graphics ■ day, dusk, and night flying modes ■ over 80 airports in four scenery areas: New York, Chicago, Los Angeles, Seattle, with additional scenery areas available ■ user-variable weather, from clear blue skies to grey cloudy conditions ■ complete flight instrumentation ■ VOR, ILS, ADF, and DME radio equipped ■ navigation facilities and course plotting ■ World War I Ace aerial battle game ■ complete information manual and flight handbook.

See your dealer . . .

or for direct orders enclose $49.95 plus $2.00 for shipping and specify UPS or first class mail delivery. American Express, Diner's Club, MasterCard, and Visa accepted.

Order Line: 800 / 637-4983

subLOGIC
Corporation
713 Edgebrook Drive
Champaign IL 61820
(217) 359-8482 Telex: 206995

Reader Service No. 146